(67577) Introduction to Machine Learning October 11, 2010

Lecture 1 – Introduction and Gentle Start

Lecturer: Shai Shalev-Shwartz *Scribe: Shai Shalev-Shwartz*

Based on a book by Shai Ben-David and Shai Shalev-Shwartz (in preparation)

Lectures 1+2 from 2009.

(67577) Introduction to Machine Learning October 18, 2010

Lecture 2 – Bias Complexity Tradeoff

Lecturer: Shai Shalev-Shwartz *Scribe: Shai Shalev-Shwartz*

Based on a book by Shai Ben-David and Shai Shalev-Shwartz (in preparation)

Lecture 3 from 2009.

(67577) Introduction to Machine Learning October 25, 2010

Lecture 3(a) – MDL

Lecturer: Shai Shalev-Shwartz *Scribe: Shai Shalev-Shwartz*

Based on a book by Shai Ben-David and Shai Shalev-Shwartz (in preparation)

Lecture 4 from 2009.

(67577) Introduction to Machine Learning October 25, 2010

Lecture 3(b) – Validation

Lecturer: Shai Shalev-Shwartz *Scribe: Shai Shalev-Shwartz*

Based on a book by Shai Ben-David and Shai Shalev-Shwartz (in preparation)

Lecture 13 from 2009.

(67577) Introduction to Machine Learning October 25, 2010

Lecture 3(c) – Compression Bounds

Lecturer: Shai Shalev-Shwartz *Scribe: Shai Shalev-Shwartz*

Based on a book by Shai Ben-David and Shai Shalev-Shwartz (in preparation)

In the previous lectures we saw how to express prior knowledge by restricting ourselves to a finite hypothesis classes or by defining an order over countable hypothesis classes. In this lecture we show how one can learn even uncountable hypothesis classes by deriving compression bounds. Roughly speaking, we shall see that if a learning algorithm can express the output hypothesis using a small subset of the training set, then the error of the hypothesis on the rest of the examples estimates its generalization error.

To motivate the results, let us first consider the following learning protocol. First, we sample a sequence of k examples denoted T. Based on these examples, we construct a hypothesis denoted h_T. Now we would like to estimate the performance of h_T so we sample a fresh sequence of $m - k$ examples, denoted V, and calculate the error of h_T on V. Since V and T are independent, we immediately get the following from Bernstein's inequality (as you saw in home exercise).

Lemma 1 $\mathbb{P}\left[L_{\mathcal{D}}(h_T) - L_V(h_T) \geq \sqrt{\frac{2L_V(h)\log(1/\delta)}{|V|}} + \frac{4\log(1/\delta)}{|V|}\right] \leq \delta$.

To derive the above bound, all we needed was independence between T and V. Therefore, we can re-define the protocol as follows. First, we agree on a sequence of k indices $I = (i_1, \ldots, i_k) \in [m]^k$. Then, we sample a sequence of m examples $S = (x_1, y_1), \ldots, (x_m, y_m)$. Now, define $T = S_I = (x_{i_1}, y_{i_1}), \ldots, (x_{i_k}, y_{i_k})$ and define V to be the rest of the examples in S. Note that this protocol is equivalent to the protocol we defined before — hence Lemma 1 still holds.

Applying a union bound over the choice of the sequence of indices we obtain the following theorem.

Theorem 1 *Let k be an integer and let $B : (\mathcal{X} \times \mathcal{Y})^k \to \mathcal{H}$ be a mapping from a sequence of k examples into a hypothesis. Let $m \geq 2k$ be a training set size and let $A : (\mathcal{X} \times \mathcal{Y})^m \to \mathcal{H}$ be a learning rule that receives a training sequence S of size m and returns a hypothesis such that $A(S) = B((x_{i_1}, y_{i_1}), \ldots, (x_{i_k}, y_{i_k}))$ for some $(i_1, \ldots, i_k) \in [m]^k$. Let $V = ((x_j, y_j) : j \notin (i_1, \ldots, i_k))$ be the sequence of examples which were not selected for defining $A(S)$. Then, with probability of at least $1 - \delta$ over the choice of S we have*

$$L_{\mathcal{D}}(A(S)) \;\leq\; L_V(A(S)) + \sqrt{L_V(A(S))\frac{4k\log(m/\delta)}{m}} + \frac{8k\log(m/\delta)}{m} \, .$$

Proof For any $I \in [m]^k$ let $h_I = B(x_{i_1}, y_{i_1}), \ldots, (x_{i_k}, y_{i_k}))$. Let $n = m - k$. Combining Lemma 1 with the union bound we have

$$\begin{aligned}
&\mathbb{P}\left[\exists I \in [m]^k \text{ s.t. } L_{\mathcal{D}}(h_I) - L_V(h_I) \geq \sqrt{\frac{2L_V(h_I)\log(1/\delta)}{n}} + \frac{4\log(1/\delta)}{n}\right] \\
&\leq \sum_{I \in [m]^k} \mathbb{P}\left[L_{\mathcal{D}}(h_I) - L_V(h_I) \geq \sqrt{\frac{2L_V(h_I)\log(1/\delta)}{n}} + \frac{4\log(1/\delta)}{n}\right] \\
&\leq m^k \delta \, .
\end{aligned}$$

Denote $\delta' = m^k\delta$. Using the assumption $k \leq 2m$ which implies that $n = m - k \geq m/2$ the above implies that with probability of at least $1 - \delta'$ we have that

$$L_{\mathcal{D}}(A(S)) \;\leq\; L_V(A(S)) + \sqrt{L_V(A(S))\frac{4k\log(m/\delta')}{m}} + \frac{8k\log(m/\delta')}{m} \, ,$$

which concludes our proof. ■

As a direct corollary we obtain:

Corollary 1 *Let k be an integer and let $B : (\mathcal{X} \times \mathcal{Y})^k \to \mathcal{H}$ be a mapping from a sequence of k examples into a hypothesis. Let $m \geq 2k$ be a training set size and let $A : (\mathcal{X} \times \mathcal{Y})^m \to \mathcal{H}$ be a learning rule that receives a training sequence S of size m and returns a hypothesis such that $A(S) = B((x_{i_1}, y_{i_1}), \ldots, (x_{i_k}, y_{i_k}))$ for some $(i_1, \ldots, i_k) \in [m]^k$. Let $V = ((x_j, y_j) : j \notin (i_1, \ldots, i_k))$ be the sequence of examples which were not selected for defining $A(S)$ and assume that $L_V(A(S)) = 0$. Then, with probability of at least $1 - \delta$ over the choice of S we have*

$$L_{\mathcal{D}}(A(S)) \;\leq\; \frac{8k\log(m/\delta)}{m} \, .$$

We can also easily derive the following:

Theorem 2 *Let k be an integer and let $B : (\mathcal{X} \times \mathcal{Y})^k \to \mathcal{H}$ be a mapping from a sequence of k examples into a hypothesis. Let $m \geq 2k$ be a training set size. Let $\delta \in (0,1)$ and assume that $m/\delta \geq 42$. Let $A : (\mathcal{X} \times \mathcal{Y})^m \to \mathcal{H}$ be a learning rule that receives a training sequence S of size m and returns a hypothesis* *such that $A(S) = B((x_{i_1}, y_{i_1}), \ldots, (x_{i_k}, y_{i_k}))$ for some $(i_1, \ldots, i_k) \in [m]^k$. Then, with probability of at least $1 - \delta$ over the choice of S we have*

$$L_{\mathcal{D}}(A(S)) \;\le\; L_S(A(S)) + \sqrt{L_S(A(S))\frac{4k\log(m/\delta)}{m}} + \frac{10k\log(m/\delta)}{m} \,.$$

Proof For any hypothesis

$$L_S(h) = \frac{|V|L_V(h) + |\bar{V}|L_{\bar{V}}(h)}{m}$$

which implies

$$L_V(h) = \frac{m}{|V|}L_S(h) - \frac{m}{|\bar{V}|}L_{\bar{V}}(h) \le \frac{m}{|V|}L_S(h) = \frac{|V| + |\bar{V}|}{|V|}L_S(h) \le L_S(h) + \frac{|\bar{V}|}{m - |\bar{V}|} \le L_S(h) + \frac{2k}{m}.$$

Combining this with the bound in Theorem 1, and using the inequality $\sqrt{a+b} \le \sqrt{a} + \sqrt{b}$ we obtain:

$$\begin{aligned}
L_{\mathcal{D}}(A(S)) \;&\le\; L_V(A(S)) + \sqrt{L_V(A(S))\frac{4k\log(m/\delta)}{m}} + \frac{8k\log(m/\delta)}{m} \\
&\le L_S(A(S)) + \sqrt{L_S(A(S))\frac{4k\log(m/\delta)}{m}} + \frac{8k\log(m/\delta)}{m} + \frac{2k}{m} + \frac{2k}{m}\sqrt{2\log(m/\delta)} \\
&= L_S(A(S)) + \sqrt{L_S(A(S))\frac{4k\log(m/\delta)}{m}} + \frac{2k}{m}\left(4\log(m/\delta) + \sqrt{2k\log(m/\delta)} + 1\right).
\end{aligned}$$

Finally, by our assumption on m/δ we have that the term in the parenthesis is at most $5\log(m/\delta)$. ■

1 Examples

For simplicity, in all the examples we assume the realizable case. At the end of this section we describe what to do in the unrealizable case.

1.1 Axis-aligned Rectangles

Note that this is an uncountable infinite class. We show that there's a simple compression scheme. Consider the algorithm A that works as follows: For each dimension, choose the two positive examples with extremal values at this dimension. Define B to be the function that returns the minimal enclosing rectangle. Then:

- $k = 2d$
- In the realizable case, $L_V(A(S)) = 0$
- Corollary 1 implies $L_{\mathcal{D}}(A(S)) \;\le\; \frac{16d\log(m/\delta)}{m}$.

1.2 Halfspaces

Let $\mathcal{X} = \mathbb{R}^d$ and consider the class $\mathbf{x} \mapsto \text{sign}(\langle \mathbf{w}, \mathbf{x} \rangle)$.

A compression scheme: W.l.o.g. assume all labels are positive (otherwise, replace $\mathbf{x}_i$ by $y_i\mathbf{x}_i$). Since the data is linearly separable, the convex hull of $\{\mathbf{x}_1, \ldots, \mathbf{x}_m\}$ does not contain the origin. Consider the point in this convex hull closest to the origin. (This is a unique point which is the Euclidean projection of the origin onto this convex hull.) Call this point $\mathbf{w}$. We claim that $\mathbf{w}$ separates the data. To see this, assume by contradiction that $\langle \mathbf{w}, \mathbf{x}_i \rangle \leq 0$ for some i. Take $\mathbf{w}' = (1-\alpha)\mathbf{w} + \alpha\mathbf{x}_i$ for $\alpha = \frac{\|\mathbf{w}\|^2}{\|\mathbf{x}_i\|^2 + \|\mathbf{w}\|^2} \in (0,1)$. Then $\mathbf{w}'$ is also in the convex hull and

$$\begin{aligned}
\|\mathbf{w}'\|^2 &= (1-\alpha)^2\|\mathbf{w}\|^2 + \alpha^2\|\mathbf{x}_i\|^2 + 2\alpha(1-\alpha)\langle \mathbf{w}, \mathbf{x}_i\rangle \\
&\leq (1-\alpha)^2\|\mathbf{w}\|^2 + \alpha^2\|\mathbf{x}_i\|^2 \\
&= \frac{\|\mathbf{x}_i\|^4\|\mathbf{w}\|^2 + \|\mathbf{x}_i\|^2\|\mathbf{w}\|^4}{(\|\mathbf{w}\|^2 + \|\mathbf{x}_i\|^2)^2} \\
&= \frac{\|\mathbf{x}_i\|^2\|\mathbf{w}\|^2}{\|\mathbf{w}\|^2 + \|\mathbf{x}_i\|^2} \\
&< \|\mathbf{w}\|^2 ,
\end{aligned}$$

which leads to a contradiction.

We have thus shown that $\mathbf{w}$ is also an ERM. Finally, since $\mathbf{w}$ is in the convex hull of the examples, we can apply Caratheodory's theorem to obtain that $\mathbf{w}$ is also in the convex hull of a subset of $d+1$ points of the polygon. Furthermore, the minimality of $\mathbf{w}$ implies that $\mathbf{w}$ must be on a face of the polygon and this implies it can be represented as a convex combination of d points.

It remains to show that $\mathbf{w}$ is also the projection onto the polygon defined by the $d+1$ points. But, this must be true: on one hand, the smaller polygon is a subset of the larger one, hence the projection onto the smaller cannot be smaller in norm. On the other hand, $\mathbf{w}$ itself is a valid solution. The uniqueness of projection concludes our proof.

1.3 Separating polynomials

Let $\mathcal{X} = \mathbb{R}^d$ and consider the class $\mathbf{x} \mapsto \text{sign}(p(x))$ where p is a degree r polynomial.

Note that $p(x)$ can be rewritten as $\langle \mathbf{w}, \psi(\mathbf{x})\rangle$ where the elements of $\psi(x)$ are all the monomials of $\mathbf{x}$ up to degree r. Therefore, the problem of constructing a comrpession scheme for $p(\mathbf{x})$ reduces to the problem of constructing a compression scheme for Halfspaces in $\mathbb{R}^{d'}$ where $d' = O(d^r)$.

1.4 What to do in the unrealizable case?

A general solution: first, find an ERM. Then, throw all the examples on which the ERM hypothesis err. Now, apply the compression scheme on the correct examples.

(67577) Introduction to Machine Learning November 1, 2010

Lecture 4 – Perceptron

Lecturer: Shai Shalev-Shwartz *Scribe: Shai Shalev-Shwartz*

Based on a book by Shai Ben-David and Shai Shalev-Shwartz (in preparation)

The class of linear separators — see Lecture 8(a) from last year. Previously, we saw a compression scheme of size d for halfspaces in $\mathbb{R}^d$. In this lecture we will introduce the Perceptron algorithm, and we will show a compression scheme which is based on *large margin* separation of the data.

2 Separation with margin

Given a training set S, we define the margin of S as

$$\gamma(S) = \max_{\mathbf{w}:\|\mathbf{w}\|_2=1} \min_{(\mathbf{x},y)\in S} y\langle \mathbf{w}, \mathbf{x}\rangle \ .$$

Geometrically, for any $\mathbf{x}$, the absolute value $|\langle \mathbf{w}, \mathbf{x}\rangle|$ is the distance between $\mathbf{x}$ and the hyperplane associated with $\mathbf{w}$. Additionally, if sign$(\langle \mathbf{w}, \mathbf{x}\rangle) = y$ then $y\langle \mathbf{w}, \mathbf{x}\rangle = |\langle \mathbf{w}, \mathbf{x}\rangle|$. Therefore, $\gamma(S)$ will be positive if S is linearly separable by a halfspace, and the value of $\gamma(S)$ captures how "fat" we can make the hyperplane while still separating the training set S.

Of course, one can increase the margin by scaling all the instances. To prevent this, we will assume from now on that the instance space is the unit ball $\mathcal{X} = \{\mathbf{x} \in \mathbb{R}^d : \|\mathbf{x}\|_2 \leq 1\}$.

An illustration is given in Figure 1.

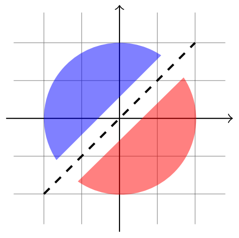


Figure 1: Separation with margin

We will later show that if $\gamma(S) > 0$ then we have a compression scheme of size at most $1/\gamma(S)^2$ for the class of Halfspaces. This is achievable by the Perceptron algorithm.

3 The Perceptron

In this section we present the Perceptron learning algorithm.

Algorithm 1 Perceptron

```
Input: A training set $S = (\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)$
Initialize: $\mathbf{w} = \mathbf{0}$
While $(\exists (\mathbf{x}, y) \in S$ s.t. $y\langle \mathbf{w}, \mathbf{x}\rangle \leq 0)$
  $\mathbf{w} = \mathbf{w} + y\mathbf{x}$
Output: $\mathbf{w}$
```

Theorem 3 *Assume that $\gamma(S) > 0$. Then, the Perceptron algorithm stops after at most $\frac{1}{\gamma(S)^2}$ iterations.*

Proof Let $\mathbf{w}^\star$ be a vector that achieves the maximum in the definition of $\gamma(S)$. Denote $\mathbf{w}_t$ to be the value of $\mathbf{w}$ at the beginning of the tth iteration of the Perceptron and by $(\mathbf{x}_t, y_t)$ the example used to update $\mathbf{w}_t$. We prove the theorem by monitoring the value of $\langle \mathbf{w}^\star, \mathbf{w}_t\rangle$. At the first iteration, $\mathbf{w}_1 = \mathbf{0}$ and therefore $\langle \mathbf{w}^\star, \mathbf{w}_1\rangle = 0$. On iteration t, we have that

$$\langle \mathbf{w}^\star, \mathbf{w}_{t+1} - \mathbf{w}_t\rangle = y\langle \mathbf{w}^\star, \mathbf{x}\rangle \geq \gamma(S) \ .$$

Therefore, after t iterations we have that $\langle \mathbf{w}^\star, \mathbf{w}_{t+1}\rangle \geq t\,\gamma(S)$. On the other hand, from Cauchy-Schwartz inequality we have that

$$\langle \mathbf{w}^\star, \mathbf{w}_{t+1}\rangle \leq \|\mathbf{w}^\star\|\,\|\mathbf{w}_{t+1}\| = \|\mathbf{w}_{t+1}\| \ .$$

Next, we upper bound the value of $\|\mathbf{w}_{t+1}\|$. Initially, $\|\mathbf{w}_1\| = 0$ and after each update we have

$$\|\mathbf{w}_{t+1}\|^2 - \|\mathbf{w}_t\|^2 = 2y_t\langle \mathbf{w}_t, \mathbf{x}_t\rangle + \|\mathbf{x}_t\|^2 \le 0 + \|\mathbf{x}_t\|^2 \le 1 \, .$$

Therefore,

$$\|\mathbf{w}_{t+1}\| \le \sqrt{t} \, . \tag{1}$$

Overall, we have shown that

$$t\,\gamma(S) \le \langle \mathbf{w}^\star, \mathbf{w}_{t+1}\rangle \le \|\mathbf{w}_{t+1}\| \le \sqrt{t} \, .$$

Rearranging the above we get that $t \le 1/\gamma(S)^2$, which concludes our proof. ■

Corollary 2 *For any training set S for which $\gamma(S) > 0$, there exists $1/\gamma(S)^2$ examples that determines a Halfspace with zero training error.*

Proof Simply run the Perceptron algorithm and note that its weight vector depends on at most $1/\gamma(S)^2$ examples. ■

Combining the above with a compression bound we obtain:

Corollary 3 *Let $\mathcal{D}$ be a distribution over $\mathcal{X} \times \mathcal{Y}$, where $\mathcal{X}$ is the unit ball of $\mathbb{R}^d$ and $\mathcal{Y} \in \{\pm 1\}$, such that there exists a unit weight vector $\mathbf{w}^\star$ and a scalar $\gamma > 0$ with*

$$\mathbb{P}_{(\mathbf{x},y)\sim\mathcal{D}}[y\langle \mathbf{w}^\star, \mathbf{x}\rangle \ge \gamma] = 1 \, .$$

Then, with probability of at least $1-\delta$ over $m \ge 2/\gamma^2$ i.i.d. examples from $\mathcal{D}$, if we run the Perceptron on the examples it will return $\mathbf{w}$ such that

$$\mathbb{P}_{(\mathbf{x},y)\sim\mathcal{D}}[y \ne \mathit{sign}(\langle \mathbf{w}, \mathbf{x}\rangle)] \le \frac{8\log(m/\delta)}{\gamma^2\, m} \, .$$

Proof Follows by combining a compression bound with the previous corollary. ■

4 The unseparable case – an inefficient solution

The analysis in the previous section holds for separable training sets. What if the data is not separable? As in the "agnostic PAC" model, we can replace separability assumption with a relative analysis, in which we compare ourselves to the number of margin errors the best $\mathbf{w}^\star$ males on the training set.

The first solution we propose is the following algorithm. The idea of the algorithm is to first find a sub-sequence of the training set of maximal size which is still separable with margin, and run the Perceptron on this sub-sequence. The immediate disadvantage of this approach is its computational complexity, which can grow like m^{1/γ^2}.

Algorithm 2 Inefficient-Perceptron for the unseparable case

```
Parameter: $\gamma > 0$
Input: A training set $S = (\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)$
Find $\mathbf{w}^\star = \mathrm{argmax}_{\mathbf{w}} |\{i : y_i\langle \mathbf{w}, \mathbf{x}_i\rangle \ge \gamma\}|$
Let $I = \{i : y_i\langle \mathbf{w}^\star, \mathbf{x}_i\rangle \ge \gamma\}$
Output: The output of Perceptron when running on the examples in $I$
```

Theorem 4 *Let $\gamma > 0$ be a margin parameter. For any $S = (\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)$, let $\mathbf{w}^\star$ be as defined in Algorithm 2 and let*

$$M_\gamma(S) = \frac{|\{i : y_i \langle \mathbf{w}^\star, \mathbf{x}_i \rangle < \gamma\}|}{m}$$

be the averaged number of margin mistakes made by $\mathbf{w}^\star$ on S. Then, with probability of at least $1 - \delta$ over a chose of $S \sim \mathcal{D}^m$, if we run the Inefficient-Perceptron algorithm its output satisfies

$$\mathop{\mathbb{P}}_{(\mathbf{x},y)\sim\mathcal{D}}[y \neq \mathit{sign}(\langle \mathbf{w}, \mathbf{x} \rangle)] \leq M_\gamma(S) + \sqrt{M_\gamma(S) \frac{4\log(m/\delta)}{\gamma^2\, m}} + \frac{8\log(m/\delta)}{\gamma^2\, m} .$$

Proof This follows directly from the general compression bound by noting that $L_V(S) \leq M_\gamma(S)$ and $k \leq 1/\gamma^2$. ■

5 The unseparable case – an efficient approach

We now describe an efficient approach. The idea is to construct a new training set which is separable and then run the original Perceptron on the new constructed training set. The i'th instance of the new constructed training set is a concatenation of a scaled version of the original instance $\mathbf{x}_i$ and a scaled version of the vector $\mathbf{e}^i$, that is, the vector which is everywhere zero except 1 in the i'th element. We denote the concatenation by $\tilde{\mathbf{x}}_i = [a\,\mathbf{x}_i\,;\ \sqrt{1-a^2}\mathbf{e}^i]$. That is, the first d elements of $\tilde{\mathbf{x}}_i$ are the vector $\mathbf{x}_i$ scaled by $a \in (0,1)$ and the rest of the elements are the vector $\mathbf{e}^i$ scaled by $\sqrt{1-a^2}$.

Algorithm 3 Efficient-Perceptron for the unseparable case

```
Parameter: $a > 0$
Input: A training set $S = (\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)$
For each $i$, let $\tilde{\mathbf{x}}_i = [a\,\mathbf{x}_i\,;\ \sqrt{1-a^2}\,\mathbf{e}^i]$
Output: Perceptron($(\tilde{\mathbf{x}}_1, y_1), \ldots, (\tilde{\mathbf{x}}_m, y_m)$)
```

Note that if $\|\mathbf{x}\| \leq 1$ then $\|\tilde{\mathbf{x}}_i\| \leq 1$ as well. In addition, trivially, the new constructed training set is separable — simply set the first d elements of $\mathbf{w}$ to zero and the rest m elements to $1/\sqrt{m}$. But, this yields a meaningless bound because the bound depends on $1/\gamma^2$ and in our case $\gamma = 1/\sqrt{m}$. The lemma below gives another interpretation to the separability of the constructed training set.

Lemma 2 *Let $\mathbf{w}^\star \in \mathbb{R}^d$ be a unit vector and let $\gamma > 0$ be a margin parameter. For each i, let*

$$\ell_i = \max\{0, \gamma - y_i \langle \mathbf{w}^\star, \mathbf{x}_i \rangle\} ,$$

denote how much is the separation with margin requirement being violated. Then, the new constructed training set $(\tilde{\mathbf{x}}_1, y_1), \ldots, (\tilde{\mathbf{x}}_m, y_m)$ is separable with margin of at least

$$\gamma' = \frac{a\,\gamma}{\sqrt{1 + \frac{a^2}{1-a^2}\sum_i \ell_i^2}}$$

Proof To simplify our notation denote $b = \sqrt{1-a^2}$ and $L = \sum_i \ell_i^2$. Set

$$\mathbf{w} = \frac{[\mathbf{w}^\star; \frac{a}{b}(y_1\ell_1, \ldots, y_m\ell_m)]}{\sqrt{1 + \frac{a^2}{b^2}L}} .$$

By construction, $\|\mathbf{w}\| = 1$. Additionally,

$$(\sqrt{1+\tfrac{a^2}{b^2}L})\, y_i\langle \mathbf{w}, \tilde{\mathbf{x}}_i\rangle = a y_i \langle \mathbf{w}^\star, \mathbf{x}_i\rangle + a\ell_i \ge a y_i\langle \mathbf{w}^\star, \mathbf{x}_i\rangle + a(\gamma - y_i\langle \mathbf{w}^\star, \mathbf{x}_i\rangle) = a\gamma \,.$$

Thus,

$$y_i\langle \mathbf{w}, \tilde{\mathbf{x}}_i\rangle \ge \frac{a\gamma}{\sqrt{1+\frac{a^2}{b^2}L}} = \frac{a\gamma}{\sqrt{1+\frac{a^2}{1-a^2}L}} \,,$$

which concludes our proof. ■

We now discuss the implications of the above lemma and the compression bound given in Corollary 1 on the performance of Algorithm 3. When running the Perceptron in the higher dimension space we know by the lemma that we need at most $B = \frac{1}{\gamma'^2}$ updates until the algorithm finds a separating hyperplane. Lets write its final vector as $[\mathbf{w}, (\xi_1, \ldots, \xi_m)]$ and note that $\mathbf{w}$ is a function of a sequence of at most B examples. Next, we show that the error of $\mathbf{w}$ on the rest of the examples is zero. Indeed, since the algorithm started with the all zeros vector, and because of the structure of the constructed examples, it follows that at most B entries of $(\xi_1, \ldots, \xi_m)$ are non-zero. For each example on which $\xi_i = 0$ we have

$$a y_i\langle \mathbf{w}, \mathbf{x}_i\rangle > 0 \quad \Rightarrow \quad y_i\langle \mathbf{w}, \mathbf{x}_i\rangle > 0 \,.$$

Denoting V the set of examples on which we didn't make an update, it follows that $L_V(\mathbf{w}) = 0$. The conditions of the compression bound given in Corollary 1 thus holds[1] and we obtain that

$$L_{\mathcal{D}}(\mathbf{w}) \le \frac{8\,B\log(\frac{m}{\delta})}{m} \,.$$

Plugging in the value of B we obtain that

$$L_{\mathcal{D}}(\mathbf{w}) \le \min_{\gamma>0}\ \min_{\mathbf{w}^\star:\|\mathbf{w}^\star\|=1} \frac{8\log(\frac{m}{\delta})}{m}\left(\frac{1}{a^2\,\gamma^2} + \frac{1}{1-a^2}\sum_{i=1}^m\left(\tfrac{1}{\gamma}\max\{0, 1 - y_i\langle \mathbf{w}^\star, \mathbf{x}_i\rangle\}\right)^2\right) \,.$$

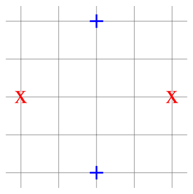
(67577) Introduction to Machine Learning November 8, 2010

Lecture 5 – Perceptron with Kernels

Lecturer: Shai Shalev-Shwartz *Scribe: Shai Shalev-Shwartz*

Based on a book by Shai Ben-David and Shai Shalev-Shwartz (in preparation)

In the previous lectures we talked about the hypothesis class of Halfspaces. Seemingly, the expressive power of Halfspaces is rather restricted – for example, it is impossible to explain the training set below by a Halfspace hypothesis.



In this lecture we present the concept of *kernels*, which makes the class of Halfspaces much more expressive. The kernel trick has had tremendous impact on machine learning theory and algorithms over the past decade.

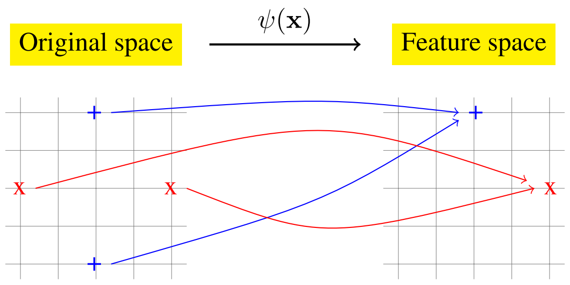
[1]Note that it is required that the size of the compression scheme, which is B in our case, will be smaller than $m/2$. But if this is not the case, the right-hand side of the bound becomes greater than 1 so it holds trivially.

6 Mapping to a feature space

To make the class of Halfspaces more expressive, we can first map the original instance space into another space (possibly, of higher dimension) and then learn a Halfspace in that space. For example, consider the example mentioned previously. Instead of learning a Halfspace in the original representation let us first define a mapping $\psi : \mathbb{R}^2 \to \mathbb{R}^2$ as follows:

$$\psi((x_1, x_2)) = (|x_1|, |x_2|) \; .$$

We use the term *feature space* to denote the range of ψ. After applying ψ the data can be easily explained using a Halfspace:



Of course, choosing a good ψ is part of our prior knowledge on the problem. But, there are some generic mappings that enable to enrich the class of Halfspaces. One notable example is polynomial mappings.

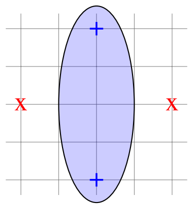
Recall that with a standard Halfspace classifier, the prediction on an instance $\mathbf{x}$ is based on the linear mapping $\mathbf{x} \mapsto \langle \mathbf{w}, \mathbf{x} \rangle$. We can generalize linear mappings to a polynomial mapping, $\mathbf{x} \mapsto p(\mathbf{x})$, where p is a polynomial of degree k. For simplicity, consider first the case in which $\mathbf{x}$ is 1 dimensional. In that case, $p(x) = \sum_{j=0}^{k} w_j x^j$, where $\mathbf{w} \in \mathbb{R}^{k+1}$ is the vector of coefficients of the polynomial we need to learn. We can rewrite $p(x) = \langle \mathbf{w}, \psi(x) \rangle$ where $\psi : \mathbb{R} \to \mathbb{R}^{k+1}$ is the mapping $x \mapsto (x^0, x^1, \ldots, x^k)$. That is, learning a k-degree polynomial in $\mathbb{R}$ can be done by learning a linear mapping in the feature space, which is $\mathbb{R}^{k+1}$ in our case.

More generally, a degree k multivariate polynomial from $\mathbb{R}^n$ to $\mathbb{R}$ can be written as

$$p(\mathbf{x}) = \sum_{J \in [n]^r : r \le k} w_J \prod_{i=1}^{r} x_{J_i} \; . \tag{2}$$

As before, we can rewrite $p(\mathbf{x}) = \langle \mathbf{w}, \psi(\mathbf{x}) \rangle$ where now $\psi : \mathbb{R}^n \to \mathbb{R}^d$ such that for each $J \in [n]^r$, $r \le k$, the coordinate of $\psi(\mathbf{x})$ associated with J is the monomial $\prod_{i=1}^{r} x_{J_i}$.

Naturally, polynomials-based classifiers yield much richer hypotheses classes than Halfspaces. For example, while the training set given in the beginning of this section cannot be explained by a Halfspace, it can be explained by an ellipse, which is a degree 2 polynomial.



So while the classifier is always linear in the feature space, it can have a highly non-linear behavior on the original space from which instances were sampled.

In general, we can choose any feature mapping ψ that maps the original instances into some *Hilbert space* (namely, a complete[2] inner-product space). The Euclidean space $\mathbb{R}^d$ is a Hilbert space for any finite d. But, there are also infinite dimensional Hilbert space (see next section).

7 The Perceptron with the kernel trick

In the previous section we saw how to enrich the class of Halfspaces by first applying a non-linear mapping, ψ, that maps the instance space into a feature space, and then learning a Halfspace in the feature space, e.g. using the Perceptron algorithm. If the range of ψ is a high dimensional space we face a computational problem since performing calculations in the high dimensional space might be too costly. In fact, even the representation of the vector $\mathbf{w}$ in the feature space can be unrealistic.

In this section we show that in order to learn a Halfspace in the feature space efficiently, it suffices to be able to efficiently calculate inner products in the feature space. Formally, let $K(\mathbf{x}, \mathbf{x}') = \langle \psi(\mathbf{x}), \psi(\mathbf{x}') \rangle$ be a function that implements inner products in the feature space. We call K a *kernel* function. Additionally, let G be an $m \times m$ matrix s.t. $G_{i,j} = K(\mathbf{x}_i, \mathbf{x}_j)$. The matrix G is often called the *Gram* matrix. We now show how to run the Perceptron algorithm in the feature space while only accessing the matrix G.

Algorithm 4 Kernelized-Perceptron

```
Input: A Gram matrix $G$, training set $S = (\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)$
Initialize: $(\alpha_1, \ldots, \alpha_m) = \mathbf{0}$
While $(\exists i \in [m]$ s.t. $y_i G\alpha \le 0)$
  $\alpha_i = \alpha_i + y_i$
Output: The predictor $h(\mathbf{x}) = \mathrm{sign}\left(\sum_{i=1}^m \alpha_i K(\mathbf{x}_i, \mathbf{x})\right)$
```

Exercise: prove that the output predictor of the Kernelized-Perceptron is the same as the output predictor of the Perceptron algorithm when running on $(\psi(\mathbf{x}_1), y_1), \ldots, (\psi(\mathbf{x}_m), y_m)$.

Example 1 (Polynomial kernels) *Consider the mapping $\psi : \mathbb{R}^n \to \mathbb{R}^d$ mapping $\mathbf{x}$ to all its monomial of order at most k. That is, for any $r \le k$ and $J \in [n]^r$ there exists a coordinate $(\psi(\mathbf{x}))_J = \prod_{i=1}^r x_{J_i}$. This is the mapping corresponds to a degree k multivariate polynomial as given in Eq. (2). To implement $\langle \psi(\mathbf{x}), \psi(\mathbf{x}') \rangle$ we note that*

$$\langle \psi(\mathbf{x}), \psi(\mathbf{x}') \rangle = \sum_{J \in [n]^r : r \le k} \left(\prod_{j=1}^r x_{J_j} \right) \left(\prod_{j=1}^r x'_{J_j} \right) = \sum_{J \in [n]^r : r \le k} \prod_{j=1}^r (x_{J_j} x'_{J_j}) = (1 + \langle \mathbf{x}, \mathbf{x}' \rangle)^k .$$

Therefore, we can learn a degree k multivariate polynomial by solving Eq. (13) with the kernel function

$$K(\mathbf{x}, \mathbf{x}') = (1 + \langle \mathbf{x}, \mathbf{x}' \rangle)^k .$$

Example 2 (Gaussian kernel) *Let the original instance space be $\mathbb{R}$ and consider the mapping ψ where for each non-negative integer $n \ge 0$ there exists an element $\psi(x)_n$ which equals to $\frac{1}{\sqrt{n!}} e^{-\frac{x^2}{2}} x^n$. Then,*

$$\langle \psi(x), \psi(x') \rangle = \sum_{n=0}^{\infty} \left(\frac{1}{\sqrt{n!}} e^{-\frac{x^2}{2}} x^n \right) \left(\frac{1}{\sqrt{n!}} e^{-\frac{(x')^2}{2}} (x')^n \right) = e^{-\frac{x^2 + (x')^2}{2}} \sum_{n=0}^{\infty} \left(\frac{(xx')^n}{n!} \right) = e^{-\frac{\|x - x'\|^2}{2}} .$$

[2] A space is complete if all Cauchy sequences in the space converge. A sequence, $\mathbf{x}_1, \mathbf{x}_2, \ldots$, in a normed space is a Cauchy sequence if for any $\epsilon > 0$ there exists a large enough n such that for any $i, j > n$ we have $\|\mathbf{x}_i - \mathbf{x}_j\| < \epsilon$. The sequence converges to a point $\mathbf{x}$ if $\|\mathbf{x}_n - x\| \to 0$ as $n \to \infty$. In our case, the norm $\|\mathbf{w}\|$ is defined by the inner product $\sqrt{\langle \mathbf{w}, \mathbf{w} \rangle}$. The reason we require the range of ψ to be in a Hilbert space is because projections in a Hilbert space are well defined. In particular, if M is a linear subset of a Hilbert space, then every $\mathbf{x}$ in the Hilbert space can be written as a sum $\mathbf{x} = \mathbf{u} + \mathbf{v}$ where $\mathbf{u} \in M$ and $\langle \mathbf{v}, \mathbf{w} \rangle = 0$ for all $\mathbf{w} \in M$. We use this fact in the proof of the Wahba's representer theorem given in the next section.

More generally, given a scalar $\sigma > 0$, the Gaussian kernel is defined to be

$$K(\mathbf{x}, \mathbf{x}') = e^{-\frac{\|\mathbf{x}-\mathbf{x}'\|^2}{2\,\sigma}} .$$

It is easy to verify that K implements an inner-product in a space in which for any n and any monomial of order n there exists an element of $\psi(\mathbf{x})$ that equals to $\frac{1}{\sqrt{n!}} e^{-\frac{\|x\|^2}{2}} \prod_{i=1}^{n} x_{j_i}$.

8 Kernels as a way to express prior knowledge*

As we have shown in the previous section, instead of specifying the feature mapping ψ, one can specify a kernel function k such that $k(\mathbf{x}, x') = \langle \psi(\mathbf{x}), \psi(\mathbf{x}') \rangle$. One can think of k as specifying similarity between instances. As an example, consider the Gaussian kernel described previously.

Given a similarity function function of the form $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$. Is it a valid kernel function? I.e. does it represent an inner product between $\mathbf{x}$ and $\mathbf{x}'$ in some feature space? The following lemma gives a sufficient (and necessary) condition.

Lemma 3 *A symmetric function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ implements an inner product in some feature space if it is positive semi-definite, namely, for all $\mathbf{x}_1, \ldots, \mathbf{x}_m$, the Gram matrix with $G_{i,j} = k(\mathbf{x}_i, \mathbf{x}_j)$ is a positive semi-definite matrix.*

Proof Define the space of functions over $\mathcal{X}$ as $\mathbb{R}^{\mathcal{X}} = \{f : \mathcal{X} \to \mathbb{R}\}$. For each $\mathbf{x} \in \mathcal{X}$ let $\psi(\mathbf{x})$ be the function $\mathbf{x} \mapsto k(\cdot, \mathbf{x})$. Define a vector space by taking all linear combinations of elements of the form $k(\cdot, \mathbf{x})$. Define an inner product on this vector space to be

$$\langle \sum_i \alpha_i k(\cdot, \mathbf{x}_i), \sum_j \beta_j k(\cdot, \mathbf{x}'_j) \rangle = \sum_{i,j} \alpha_i \beta_j k(\mathbf{x}_i, \mathbf{x}'_j) .$$

This is a valid inner product since it is symmetric (because k is symmetric), it is linear (immediate), and it is positive semi-definite. Clearly,

$$\langle \psi(\mathbf{x}), \psi(\mathbf{x}') \rangle = \langle k(\cdot, \mathbf{x}), k(\cdot, \mathbf{x}) \rangle = k(\mathbf{x}, \mathbf{x}') ,$$

which concludes our proof. ■

(67577) Introduction to Machine Learning November 15, 2010

Lecture 6 – Convex Learning Problems

Lecturer: Shai Shalev-Shwartz *Scribe: Shai Shalev-Shwartz*

Based on a book by Shai Ben-David and Shai Shalev-Shwartz (in preparation)

In this lecture we will focus on convex learning problems. This is an important class of learning problems, mainly because most of what we can learn efficiently falls into this class. Furthermore, in many cases we use surrogate convex loss functions (instead of the original loss function) to facilitate an efficient solution.

9 Convex learning problems

In convex learning problems, each hypothesis is parameterized by a vector $\mathbf{w} \in W$ and the loss function can be written as (with slight abuse of notation)

$$\ell(h_{\mathbf{w}}(\mathbf{x}), y) = \ell(\mathbf{w}, (\mathbf{x}, y)) ,$$

where $\ell(\mathbf{w}, \mathbf{z})$ is convex with respect to $\mathbf{w}$. To simplify the notation throughout this lecture, we use $\mathbf{z}$ as a shorthand for $(\mathbf{x}, y)$. The learning problem is to minimize

$$L_{\mathcal{D}}(\mathbf{w}) \; = \; \mathop{\mathbb{E}}_{\mathbf{z} \sim \mathcal{D}} \ell(\mathbf{w}, \mathbf{z}) .$$

As in previous lectures, we only have an access to $\mathcal{D}$ by sampling i.i.d. examples out of it and we would like to find an ϵ-accurate solution with high probability over the examples. For simplicity, throughout the section we will only analyze the expected accuracy of our solutions, but high probability bounds can also be obtained.

Example 3 (Linear regression with squared loss) *Suppose that $h_{\mathbf{w}}(\mathbf{x}) = \langle \mathbf{w}, \mathbf{x} \rangle$ and $\ell(h_{\mathbf{w}}(\mathbf{x}), y) = (h_{\mathbf{w}}(\mathbf{x}) - y)^2$. Then, set $\mathbf{z} = (\mathbf{x}, y)$ and $\ell(\mathbf{w}, \mathbf{z}) = (\langle \mathbf{w}, \mathbf{x} \rangle - y)^2$. This loss function is convex w.r.t. $\mathbf{w}$.*

10 Surrogate loss functions

In previous lectures we mainly dealt with binary classification with the $0-1$ loss. For the class of Halfspaces, i.e. $\mathbf{x} \mapsto \text{sign}(\langle \mathbf{w}, \mathbf{x} \rangle)$, this amounts to the loss function (w.r.t. $\mathbf{w}$)

$$\ell_{0-1}(\mathbf{w}, \mathbf{z}) = \mathbb{1}_{[y \langle \mathbf{w}, \mathbf{x} \rangle \leq 0]} .$$

This loss function is not convex w.r.t. $\mathbf{w}$ and indeed solving the ERM problem w.r.t. the $0-1$ loss in the unrealizable case is known to be NP-hard.

To circumvent the hardness result, one popular approach is to upper bound the non-convex loss function by a convex surrogate loss function. For example,

$$\ell_{0-1}(\mathbf{w}, \mathbf{z}) \leq \ell_{\text{hinge}}(\mathbf{w}, \mathbf{z}) \stackrel{\text{def}}{=} \max\{0, 1 - y\langle \mathbf{w}, \mathbf{x} \rangle\} .$$

An illustration of the functions ℓ_{0-1} and ℓ_{hinge} is given in Figure 2.

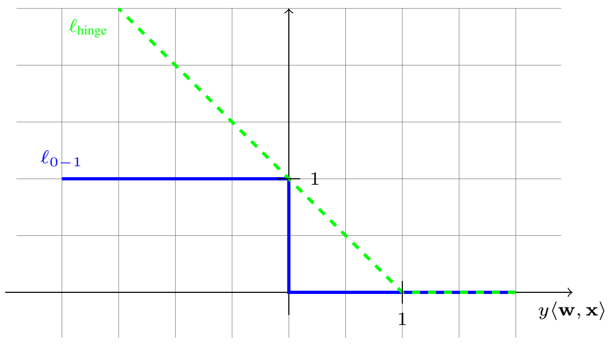


Figure 2: An illustration of the $0-1$ loss and the hinge-loss as scalar functions of $y\langle \mathbf{w}, \mathbf{x} \rangle$.

In the next sections we discuss several simple generic methods for solving convex learning problems. But, first, we recall basic definitions from convex analysis.

11 Convexity

A set A is convex if for any two vectors $\mathbf{w}_1, \mathbf{w}_2$ in A, all the line between $\mathbf{w}_1$ and $\mathbf{w}_2$ is also within A. That is, for any $\alpha \in [0,1]$ we have that $\alpha \mathbf{w}_1 + (1-\alpha)\mathbf{w}_2 \in A$. A function $f : A \to \mathbb{R}$ is convex if for all $\mathbf{u}, \mathbf{v} \in \mathbb{R}^n$ and $\alpha \in [0,1]$ we have

$$f(\alpha \mathbf{u} + (1-\alpha)\mathbf{v}) \ \le \ \alpha f(\mathbf{u}) + (1-\alpha) f(\mathbf{v}) \ .$$

It is easy to verify that f is convex iff its *epigraph* is a convex set, where $\mathrm{epigraph(f)} = \{(\mathbf{x}, \alpha) : f(\mathbf{x}) \le \alpha\}$. Throughout this lecture we focus on convex functions.

Gradient: The gradient of a differential function $f : \mathbb{R}^d \to \mathbb{R}$ at $\mathbf{w}$ is the vector of partial derivatives of f w.r.t. its elements, namely, $\nabla f(\mathbf{w}) = \left(\frac{\partial f(\mathbf{w})}{\partial w_1}, \ldots, \frac{\partial f(\mathbf{w})}{\partial w_d}\right)$. The gradient of f at $\mathbf{w}$ yields the first order Taylor approximation of f around $\mathbf{w}$ by $f(\mathbf{u}) \approx f(\mathbf{w}) + \langle \mathbf{u} - \mathbf{w}, \nabla f(\mathbf{w}) \rangle$. When f is convex, this approximation lower bounds f, that is,

$$f(\mathbf{u}) \ge f(\mathbf{w}) + \langle \mathbf{u} - \mathbf{w}, \nabla f(\mathbf{w}) \rangle \ .$$

Sub-gradients: A vector $\boldsymbol{\lambda}$ is a *sub-gradient* of a function f at $\mathbf{w}$ if for all $\mathbf{u} \in A$ we have that

$$f(\mathbf{u}) - f(\mathbf{w}) \ge \langle \mathbf{u} - \mathbf{w}, \boldsymbol{\lambda} \rangle \ .$$

The *differential set* of f at $\mathbf{w}$, denoted $\partial f(\mathbf{w})$, is the set of all sub-gradients of f at $\mathbf{w}$. For scalar functions, a sub-gradient of a convex function f at x is a slope of a line that touches f at x and is not above f everywhere.

Two useful properties of sub-gradients are given below:

1. If f is differentiable at $\mathbf{w}$ then $\partial f(\mathbf{w})$ consists of a single vector which amounts to the *gradient* of f at $\mathbf{w}$ and is denoted by $\nabla f(\mathbf{w})$. In this sense, the notion of sub-gradient generalizes the notion of gradient.
2. If $g(\mathbf{w}) = \max_{i \in [r]} g_i(\mathbf{w})$ for r convex differentiable functions $g_1, \ldots, g_r$, and $j = \arg\max_i g_i(\mathbf{u})$, then the gradient of g_j at $\mathbf{u}$ is a subgradient of g at $\mathbf{u}$.

The following lemma gives useful ways to know if a function is convex:

Lemma 4 *A function f is convex if one of the following holds*

1. *f is differentiable and all its gradient at each point is also a sub-gradient.*
2. *f is twice differentiable and its Hessian (matrix of second derivatives) at each point is positive semi-definite.*
3. *f is a composition of a convex scalar function on a linear vector function.*
4. *f is a maximum of linear functions.*

Proof

1. Follows immidiatly from the relation between the gradient and convexity defined above.
2. If $f : R^d \to R$ is twice differentiable then for every 2 points $u, w \in R^d$ we can write :

$$f(u) = f(\mathbf{w}) + \langle \mathbf{u} - \mathbf{w}, \nabla f(\mathbf{w}) \rangle + (\mathbf{u} - \mathbf{w})^T \mathcal{H}_f(\xi)(\mathbf{u} - \mathbf{w})$$

where ξ is some convex combination of u and w.
But since $\mathcal{H}_f$ is PSD then the right most expression is ≥ 0 and thus f(u) is always bigger than its first order taylor approximation and thus f is convex.

3. If $f(w) = g(\langle w, x\rangle)$ where g is convex, then :

$$f(\alpha w + (1-\alpha)u) = g(\alpha\langle w, x\rangle + (1-\alpha)\langle u, x\rangle)$$

   from the linearity of the inner product.
   From the convexity of g we get :

$$\leq \alpha g(\langle w, x\rangle) + (1-\alpha)g(\langle u, x\rangle) = \alpha f(w) + (1-\alpha)f(u)$$

   as required.

4. If $f(\mathbf{w}) = \max_{i\in[r]} g_i(\mathbf{w})$ for r linear functions $g_1, \ldots, g_r$ (i.e. $g_i(w) = \langle w, x_i\rangle$), then :

$$f(\alpha w + (1-\alpha)u) = \max_{i\in[r]} g_i(\alpha\mathbf{w} + (1-\alpha)u) = \max_{i\in[r]}\langle\alpha\mathbf{w} + (1-\alpha)u, x_i\rangle$$

   If we denote by x^* the function where this expression obtains the maximum, than from the linearity of the inner product we get :

$$= \alpha\langle w, x^*\rangle + (1-\alpha)\langle u, x^*\rangle \leq \alpha\max_{i\in[r]}\langle w, x_i\rangle + (1-\alpha)\max_{i\in[r]}\langle u, x_i\rangle = \alpha f(w) + (1-\alpha)f(u)$$

   as required.

■

Example 4 (Logistic-loss) *Recall that the logistic-loss is defined as $\ell(\mathbf{w}; \mathbf{x}, y) = \log(1 + e^{-y\langle\mathbf{w},\mathbf{x}\rangle})$. This function is a composition of the scalar function $\log(1 + e^{-ax})$ with the linear function $\mathbf{w} \mapsto y\langle\mathbf{w}, \mathbf{x}\rangle$. Calculating the derivatives of the scalar function we get :*

$$\ell'(x) = \frac{e^{-ax} * -a}{1 + e^{-ax}} = \frac{\frac{-a}{e^{ax}}}{\frac{e^{ax}+1}{e^{ax}}} = \frac{-a}{e^{ax} + 1}$$

and :

$$\ell''(x) = \frac{a}{(e^{ax} + 1)^2} * e^{ax}a \geq 0$$

hence ℓ is convex. Since this function is differentiable, a sub-gradient at $\mathbf{w}$ is the gradient at $\mathbf{w}$, which using the chain rule equals to

$$\nabla\ell(\mathbf{w}; \mathbf{x}, y) = \frac{-\exp(-y\langle\mathbf{w}, \mathbf{x}\rangle)}{1 + \exp(-y\langle\mathbf{w}, \mathbf{x}\rangle)}\, y\,\mathbf{x} = \frac{-1}{1 + \exp(y\langle\mathbf{w}, \mathbf{x}\rangle)}\, y\,\mathbf{x}\ .$$

Example 5 (Hinge-loss) *Recall that the hinge-loss is defined as $\ell(\mathbf{w}; \mathbf{x}, y) = \max\{0, 1 - y\langle\mathbf{w}, \mathbf{x}\rangle\}$. This is the maximum of two linear functions, hence it is convex. Additionally, using the two properties above of sub-gradients we have that if $1 - y\langle\mathbf{w}, \mathbf{x}\rangle > 0$ then $-y\,\mathbf{x} \in \partial\ell(\mathbf{w}; \mathbf{x}, y)$ and if $1 - y\langle\mathbf{w}, \mathbf{x}\rangle < 0$ then $\mathbf{0} \in \partial\ell(\mathbf{w}; \mathbf{x}, y)$. Furthermore, it is easy to verify that*

$$\partial\ell(\mathbf{w}; \mathbf{x}, y) = \begin{cases} \{-y\mathbf{x}\} & \textit{if } 1 - y\langle\mathbf{w}, \mathbf{x}\rangle > 0 \\ \{\mathbf{0}\} & \textit{if } 1 - y\langle\mathbf{w}, \mathbf{x}\rangle < 0 \\ \{-\alpha y\mathbf{x} : \alpha \in [0, 1]\} & \textit{if } 1 - y\langle\mathbf{w}, \mathbf{x}\rangle = 0 \end{cases}$$

the last case can be proved as follows :
Because $\ell(w; x, y) = 0$, so :

$$\ell(u; x, y) - \ell(w; x, y) = \max\{0, 1 - y\langle\mathbf{u}, \mathbf{x}\rangle\}$$

Because $y\langle w, x\rangle = 1$, for every u,

$$\langle u - w, -\alpha y x\rangle = -\alpha(y\langle u, x\rangle - y\langle w, x\rangle) = \alpha(1 - y\langle u, x\rangle)$$

but simply :

$$\max\{0, 1 - y\langle \mathbf{u}, \mathbf{x}\rangle\} \geq \alpha(1 - y\langle u, x\rangle)$$

since $\alpha \in (0, 1)$ so we get

$$\ell(u; x, y) - \ell(w; x, y) \geq \langle u - w, -\alpha y x\rangle$$

which is the condition for the subgradient.

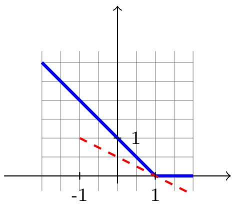


Figure 3: An illustration of the hinge-loss function $f(x) = \max\{0, 1 - x\}$ and one of its sub-gradients at $x = 1$.

Lipschitz functions: We say that $f : A \to \mathbb{R}$ is ρ-Lipschitz if for all $\mathbf{u}, \mathbf{v} \in A$

$$|f(\mathbf{u}) - f(\mathbf{v})| \leq \rho \|\mathbf{u} - \mathbf{v}\| \ .$$

An equivalent definition is that the ℓ_2 norm of all sub-gradients of f at points in A is bounded by ρ.

12 Gradient Descent

Let $f(\mathbf{w})$ be a convex differentiable function which we would like to minimize. One of the simplest optimization methods is called gradient descent. Gradient descent is an iterative method in which we start with an initial vector $\mathbf{w}_1$ and on each iteration we update $\mathbf{w}_{t+1} = \mathbf{w}_t - \eta_t \nabla f(\mathbf{w}_t)$, where η_t is a scalar called "step-size". That is, we move from $\mathbf{w}_t$ in the opposite direction of the gradient. The reason to update using the direction of the gradient is because locally, this is the best direction to take (recall the definition of derivatives).

In the context of this lecture, our goal is to minimize the function $L_{\mathcal{D}}(\mathbf{w})$. We cannot use the gradient descent approach since we do not know how to calculate the gradient of $L_{\mathcal{D}}(\mathbf{w})$ (since $\mathcal{D}$ is unknown).

13 Stochastic Gradient Descent

Similarly to gradient descent, *stochastic* gradient descent is an iterative method. The difference is that the update is not based on the gradient of the function we would like to minimize but rather we update $\mathbf{w}_{t+1} = w_t - \eta_t \mathbf{v}_t$, where $\mathbf{v}_t$ is a random vector whose expected value is $\nabla f(\mathbf{w}_t)$ (or more generally, its expected value is a sub-gradient of f at $\mathbf{w}_t$).

The importance of stochastic gradient descent in our context stems from the fact that we can easily apply it to the function $L_{\mathcal{D}}(\mathbf{w})$ as the following lemma shows.

Lemma 5 *Let $L_{\mathcal{D}}(\mathbf{w}) = \mathbb{E}_{\mathbf{z}\sim\mathcal{D}}[\ell(\mathbf{w}, \mathbf{z})]$. For any $\mathbf{w}$, define $\boldsymbol{\lambda}$ by first sampling $\mathbf{z} \sim \mathcal{D}$ and then setting $\boldsymbol{\lambda} \in \partial\ell(\mathbf{w}, \mathbf{z})$. Then, $\mathbb{E}_{\mathbf{z}}[\boldsymbol{\lambda}] \in \partial L_{\mathcal{D}}(\mathbf{w})$.*

Proof By definition of $\boldsymbol{\lambda}$ we have that for all $\mathbf{z}$ and all $\mathbf{u}$, $\ell(\mathbf{u},\mathbf{z}) - \ell(\mathbf{w},z) \geq \langle \mathbf{u}-\mathbf{w}, \boldsymbol{\lambda}\rangle$. Taking expectation w.r.t. the choise of $\mathbf{z}$ on both sides we get that

$$\mathop{\mathbb{E}}_{\mathbf{z}\sim\mathcal{D}}[\langle \mathbf{u}-\mathbf{w},\boldsymbol{\lambda}\rangle] \leq \mathop{\mathbb{E}}_{\mathbf{z}\sim\mathcal{D}}[\ell(\mathbf{u},\mathbf{z}) - \ell(\mathbf{w},\mathbf{z})]$$

which yields

$$\langle \mathbf{u}-\mathbf{w}, \mathop{\mathbb{E}}_{\mathbf{z}}[\boldsymbol{\lambda}]\rangle \leq L_{\mathcal{D}}(\mathbf{u}) - L_{\mathcal{D}}(\mathbf{w}) \ .$$

But, this implies that $\mathbb{E}_{\mathbf{z}}[\boldsymbol{\lambda}]$ is a sub-gradient of $L_{\mathcal{D}}(\mathbf{w})$ as required. ■

We next describe three variants of stochastic gradient descent.

13.1 Convex-Lipschitz functions

Algorithm 5 Stochastic sub-gradient descent (for Lipschitz-convex functions)

```
**Goal:** Solve $\min_{\mathbf{w}\in A} f(\mathbf{w})$
**Initialize:** $\mathbf{w}_1 = \mathbf{0}$ ; Choose $\eta_1 \in \mathbb{R}$
**for** $t = 1,\ldots,T$
  Choose random vector $\mathbf{v}_t$ s.t. $\mathbb{E}[\mathbf{v}_t] \in \partial f(\mathbf{w}_t)$
  Set $\eta_t = \eta_1/\sqrt{t}$
  Set $\mathbf{w}'_t = \mathbf{w}_t - \eta_t \mathbf{v}_t$
  Set $\mathbf{w}_{t+1} = \arg\min_{\mathbf{w}\in A} \|\mathbf{w} - \mathbf{w}'_t\|^2$
**end for**
**Output:** $\bar{\mathbf{w}} = \frac{1}{T}\sum_{t=1}^{T} \mathbf{w}_t$
```

We now analyze the convergence of Algorithm 5 and show that the expected value of the objective at the output of Algorithm 5 converges to the optimal value.

Theorem 5 *Assume that f is convex and that $\mathbb{E}[\|\mathbf{v}_t\|^2] \leq \rho^2$. Let $\mathbf{u} \in \operatorname{argmin}_{\mathbf{w}\in A} f(\mathbf{w})$ be an optimal solution and let U be s.t. $\sup\{\|\mathbf{w}-\mathbf{u}\| : \mathbf{w}\in A\} \leq U$. Then,*

$$\mathbb{E}[f(\bar{\mathbf{w}})] - f(\mathbf{u}) \ \leq \ \frac{1}{\sqrt{T}}\left(\frac{U^2}{2\eta_1} + \rho^2\,\eta_1\right) ,$$

where expectation is w.r.t. the randomness in choosing $\mathbf{v}_1,\ldots,\mathbf{v}_T$. In particular, setting $\eta_1 = \frac{U}{\rho\sqrt{2}}$ gives

$$\mathbb{E}[f(\bar{\mathbf{w}})] - f(\mathbf{u}) \ \leq \ U\rho\sqrt{\frac{2}{T}} \ .$$

Proof According to Jensen's inequatlity :

$$f(\bar{\mathbf{w}}) \leq \tfrac{1}{T}\sum_{t=1}^{T} f(\mathbf{w}_t)$$

Taking expectations on both sides we get :

$$\mathbb{E}[f(\bar{\mathbf{w}})] \leq \mathbb{E}[\tfrac{1}{T}\sum_{t=1}^{T} f(\mathbf{w}_t)] = \tfrac{1}{T}\sum_{t=1}^{T}\mathbb{E}[f(\mathbf{w}_t)] \ .$$

So, it suffices to prove that :

$$\tfrac{1}{T}\sum_{t=1}^{T}\mathbb{E}[f(\mathbf{w}_t)] - f(\mathbf{u}) \le \frac{1}{\sqrt{T}}\left(\frac{U^2}{2\eta_1} + \rho^2\,\eta_1\right)$$

STEP 1 : If for every t :

$$\mathbb{E}[f(\mathbf{w}_t)] - f(\mathbf{u}) \quad \le \quad \mathbb{E}\left[\frac{\|\mathbf{w}_t - \mathbf{u}\|^2 - \|\mathbf{w}_{t+1} - \mathbf{u}\|^2}{2\eta_t}\right] + \frac{\eta_t}{2}\,\rho^2 \,.$$

then the bound holds.

PROOF STEP 1 :

Summing the former equation over t and rearranging we obtain

$$\sum_{t=1}^{T}\left(\mathbb{E}[f(\mathbf{w}_t)] - f(\mathbf{u})\right) \quad \le \quad \mathbb{E}\left[\|\mathbf{w}_1 - \mathbf{u}\|^2\tfrac{1}{2\eta_1} + \sum_{t=2}^{T}\|\mathbf{w}_t - \mathbf{u}\|^2\left(\tfrac{1}{2\eta_t} - \tfrac{1}{2\eta_{t-1}}\right) - \|\mathbf{w}_{T+1} - \mathbf{u}\|^2\tfrac{1}{2\eta_T}\right] + \tfrac{\rho^2}{2}\sum_{t=1}^{T}\eta_t$$

Since $(-\|\mathbf{w}_{T+1} - \mathbf{u}\|^2\frac{1}{2\eta_T}) \le 0$ we can drop it from the right hand side. Additionally, for every t, $\|\mathbf{w}_t - \mathbf{u}\|^2 \le U^2$, and thus

$$\le \mathbb{E}\left[U^2\left(\tfrac{1}{2\eta_1} + \sum_{t=2}^{T}\left(\tfrac{1}{2\eta_t} - \tfrac{1}{2\eta_{t-1}}\right)\right)\right] + \tfrac{\rho^2}{2}\sum_{t=1}^{T}\eta_t = U^2\left(\tfrac{1}{2\eta_1} + \sum_{t=2}^{T}\left(\tfrac{1}{2\eta_t} - \tfrac{1}{2\eta_{t-1}}\right)\right) + \tfrac{\rho^2}{2}\sum_{t=1}^{T}\eta_t$$

Canceling out terms and using the definition $\eta_t = \frac{\eta_1}{\sqrt{t}}$ we get :

$$= U^2\tfrac{1}{2\eta_T} + \tfrac{\rho^2\,\eta_1}{2}\sum_{t=1}^{T}\tfrac{1}{\sqrt{t}} \le U^2\tfrac{\sqrt{T}}{2\eta_1} + \rho^2\,\eta_1\sqrt{T} \,,$$

where in the last transition we used

$$\sum_{t=1}^{T}\tfrac{1}{\sqrt{t}} \le \int_0^T\frac{dx}{\sqrt{x}} = 2\sqrt{T}$$

Dividing both sides of the above by T we get the proof of STEP 1.

STEP 2 : For every t :

$$\mathbb{E}[f(\mathbf{w}_t)] - f(\mathbf{u}) \quad \le \quad \mathbb{E}\left[\frac{\|\mathbf{w}_t - \mathbf{u}\|^2 - \|\mathbf{w}_{t+1} - \mathbf{u}\|^2}{2\eta_t}\right] + \frac{\eta_t}{2}\,\rho^2 \,.$$

PROOF OF STEP 2:

We start with the following lemma.

Lemma 6 (Projection lemma) *Let A be a convex set, let $\mathbf{u} \in A$, and let $\mathbf{v}$ be the projection of $\mathbf{w}$ on A, i.e.*

$$\mathbf{v} = \operatorname*{argmin}_{\mathbf{x}\in A}\|\mathbf{w} - \mathbf{x}\|^2 \,.$$

Then,

$$\|\mathbf{w} - \mathbf{u}\|^2 - \|\mathbf{v} - \mathbf{u}\|^2 \ge 0 \,.$$

Proof Since the desired inequality measures relative distances between $\mathbf{w}, \mathbf{v}, \mathbf{u}$ we can translate everything so that $\mathbf{v}$ will be the zero vector. If $\mathbf{w} \in A$ then the claim is trivial. Otherwise, the gradient of the objective of the optimization problem in the definition of $\mathbf{v}$ must point inside the set A at the point v (the objective function gets larger when moving inwards in A). Formally,

$$\langle -(\mathbf{w}-\mathbf{v}), \mathbf{u}-\mathbf{v}\rangle \geq 0 \, .$$

i.e. the angle between the gradient and any vector pointing from v to a point in A should be $\in [-90^0, 90^0]$. This condition is true localy for every differentiable function and is true globaly for convex functions. Thus,

$$\|\mathbf{w}-\mathbf{u}\|^2 - \|\mathbf{v}-\mathbf{u}\|^2 = \|\mathbf{w}\|^2 - \|\mathbf{v}\|^2 - 2\langle \mathbf{w}-\mathbf{v}, \mathbf{u}\rangle = \|\mathbf{w}\|^2 - \|\mathbf{v}\|^2 + 2\langle -(\mathbf{w}-\mathbf{v}), \mathbf{u}-\mathbf{v}\rangle$$

Where the last transition is becasause $v = \vec{0}$ and the linearity of the inner product. Now if we use the identity above we get :

$$\geq \|\mathbf{w}\|^2 - \|\mathbf{v}\|^2 = \|\mathbf{w}\|^2 \geq 0 \, .$$

Where again the last transition is because $v = \vec{0}$. ■

Now, we shall prove STEP 2 by analyzing the potential $\mathbb{E}[\|\mathbf{w}_t - \mathbf{u}\|^2]$. Initially, $\|\mathbf{w}_1 - \mathbf{u}\|^2 = \|\mathbf{u}\|^2$. Additionally,

$$\|\mathbf{w}_t - \mathbf{u}\|^2 - \|\mathbf{w}_{t+1} - \mathbf{u}\|^2 \;=\; \left(\|\mathbf{w}_t - \mathbf{u}\|^2 - \|\mathbf{w}'_t - \mathbf{u}\|^2\right) + \left(\|\mathbf{w}'_t - \mathbf{u}\|^2 - \|\mathbf{w}_{t+1} - \mathbf{u}\|^2\right) \, .$$

The second summand is non-negative because of the projection lemma ($\mathbf{w}_{t+1}$ is the projection "v" in the lemma). For the first summand, we plugin the definition of $\mathbf{w}'_t$ to get that

$$\|\mathbf{w}_t - \mathbf{u}\|^2 - \|\mathbf{w}'_t - \mathbf{u}\|^2 = 2\eta_t\langle \mathbf{w}_t - \mathbf{u}, \mathbf{v}_t\rangle - \eta_t^2\|\mathbf{v}_t\|^2 \, .$$

Taking expectation of both sides and using the law of total expectation we get

$$\mathbb{E}[\|\mathbf{w}_t - \mathbf{u}\|^2 - \|\mathbf{w}'_t - \mathbf{u}\|^2] = \mathop{\mathbb{E}}_{\mathbf{v}_1,\ldots,\mathbf{v}_{t-1}}[2\eta_t\langle \mathbf{w}_t - \mathbf{u}, \mathop{\mathbb{E}}_{\mathbf{v}_t}[\mathbf{v}_t]\rangle] - \eta_t^2\,\mathbb{E}[\|\mathbf{v}_t\|^2] \, .$$

The fact that $\mathbb{E}_{\mathbf{v}_t}[\mathbf{v}_t]$ is a sub-gradient of f at $\mathbf{w}_t$ implies that $\langle \mathbf{w}_t - \mathbf{u}, \mathbb{E}[\mathbf{v}_t]\rangle \geq f(\mathbf{w}_t) - f(\mathbf{u})$. Therefore, combining also the assumption $\mathbb{E}[\|\mathbf{v}_t\|^2] \leq \rho^2$ we get :

$$\mathbb{E}[\|\mathbf{w}_t - \mathbf{u}\|^2 - \|\mathbf{w}_{t+1} - \mathbf{u}\|^2] \quad\geq\quad 2\eta_t(\mathbb{E}[f(\mathbf{w}_t)] - f(\mathbf{u})) - \eta_t^2\,\rho^2 \, .$$

Rearranging the above gives

$$\mathbb{E}[f(\mathbf{w}_t)] - f(\mathbf{u}) \quad\leq\quad \mathbb{E}\left[\frac{\|\mathbf{w}_t - \mathbf{u}\|^2 - \|\mathbf{w}_{t+1} - \mathbf{u}\|^2}{2\eta_t}\right] + \frac{\eta_t}{2}\,\rho^2 \, .$$

■

As a corollary we obtain:

Corollary 4 *Assume we run Algorithm 5 with $f(\mathbf{w}) = L_{\mathcal{D}}(\mathbf{w})$ and produces $\mathbf{v}_t$ as in Lemma 5. Assume that $\ell(\mathbf{w}, \mathbf{z})$ is convex and ρ-Lipschitz w.r.t. its first argument. Then,*

$$\mathbb{E}[L_{\mathcal{D}}(\bar{\mathbf{w}})] - L_{\mathcal{D}}(\mathbf{u}) \;\leq\; U\rho\sqrt{\frac{2}{T}} \, .$$

The above corollary gives us a learning algorithm for any convex-Lipschitz loss function. In particular, by noting that the number of iterations is the number of examples we sample from the distribution, the corollary implies that the sample complexity for learning the problem is

$$m \geq \frac{2\rho^2 U^2}{\epsilon^2} \, .$$

13.2 Strongly-convex functions

In this section we show a faster convergence rate (which also implies a lower sample complexity) for learning strongly-convex function. A function f is λ-strongly convex if

$$f(\alpha \mathbf{u} + (1-\alpha)\mathbf{v}) \ge \alpha f(\mathbf{u}) + (1-\alpha) f(\mathbf{v}) + \frac{\alpha(1-\alpha)\lambda}{2}\|\mathbf{u}-\mathbf{v}\|^2 .$$

This definition implies that for any $\mathbf{w}, \mathbf{u}$ and $\mathbf{v} \in \partial f(\mathbf{w})$ we have

$$\langle \mathbf{w}-\mathbf{u}, \mathbf{v}\rangle \ \ge \ f(\mathbf{w}) - f(\mathbf{u}) + \tfrac{\lambda}{2}\|\mathbf{w}-\mathbf{u}\|^2 .$$

Algorithm 6 Stochastic sub-gradient descent (for strongly-convex functions)

```
Goal: Solve $\min_{\mathbf{w}\in A} f(\mathbf{w})$
Initialize: $\mathbf{w}_1 = \mathbf{0}$ ; Strong-convexity parameter $\lambda$
for $t = 1, \ldots, T$
  Choose random vector $\mathbf{v}_t$ s.t. $\mathbb{E}[\mathbf{v}_t] \in \partial f(\mathbf{w}_t)$
  Set $\eta_t = \frac{1}{\lambda t}$
  Set $\mathbf{w}'_t = \mathbf{w}_t - \eta_t \mathbf{v}_t$
  Set $\mathbf{w}_{t+1} = \arg\min_{\mathbf{w}\in A} \|\mathbf{w}-\mathbf{w}'_t\|^2$
end for
Output: $\bar{\mathbf{w}} = \frac{1}{T}\sum_{t=1}^{T} \mathbf{w}_t$
```

Lemma 7 *Let f be a λ-strongly convex function. Assume that f is λ-strongly-convex and that $\mathbb{E}[\|\mathbf{v}_t\|^2] \le \rho^2$. Let $\mathbf{u} \in \mathrm{argmin}_{\mathbf{w}\in A} f(\mathbf{w})$ be an optimal solution. Then,*

$$\mathbb{E}[f(\bar{\mathbf{w}})] - f(\mathbf{u}) \le \frac{\rho^2}{2\,\lambda\, T}(1 + \ln(T)) .$$

Proof Let $\nabla_t = \mathbb{E}[\mathbf{v}_t]$. Since f is strongly convex and ∇_t is in the sub-gradient set of f at $\mathbf{w}_t$ we have that

$$\langle \mathbf{w}_t - \mathbf{u}, \nabla_t\rangle \ \ge \ f(\mathbf{w}_t) - f(\mathbf{u}) + \tfrac{\lambda}{2}\|\mathbf{w}_t - \mathbf{u}\|^2 . \tag{3}$$

Next, we show that

$$\langle \mathbf{w}_t - \mathbf{u}, \nabla_t\rangle \ \le \ \frac{\mathbb{E}[\|\mathbf{w}_t-\mathbf{u}\|^2 - \|\mathbf{w}_{t+1}-\mathbf{u}\|^2]}{2\,\eta_t} + \tfrac{\eta_t}{2}\rho^2 . \tag{4}$$

Since $\mathbf{w}_{t+1}$ is the projection of $\mathbf{w}'_t$ onto A, and $\mathbf{u} \in A$ we have that $\|\mathbf{w}'_t - \mathbf{u}\|^2 \ge \|\mathbf{w}_{t+1}-\mathbf{u}\|^2$. Therefore,

$$\begin{aligned}\|\mathbf{w}_t-\mathbf{u}\|^2 - \|\mathbf{w}_{t+1}-\mathbf{u}\|^2 \ &\ge \ \|\mathbf{w}_t-\mathbf{u}\|^2 - \|\mathbf{w}'_t-\mathbf{u}\|^2 \\ &= \ 2\eta_t\langle \mathbf{w}_t-\mathbf{u}, \mathbf{v}_t\rangle - \eta_t^2\|\mathbf{v}_t\|^2 \ .\end{aligned}$$

Taking expectation of both sides, rearranging, and using the assumption $\mathbb{E}[\|\mathbf{v}_t\|^2] \le \rho^2$ yields Eq. (4). Comparing Eq. (3) and Eq. (4) and summing over t we obtain

$$\sum_{t=1}^{T}(\mathbb{E}[f(\mathbf{w}_t)] - f(\mathbf{u})) \ \le$$

$$\mathbb{E}\left[\sum_{t=1}^{T}\left(\frac{\|\mathbf{w}_t-\mathbf{u}\|^2 - \|\mathbf{w}_{t+1}-\mathbf{u}\|^2}{2\,\eta_t} + \tfrac{\lambda}{2}\|\mathbf{w}_t-\mathbf{u}\|^2\right)\right] + \frac{\rho^2}{2}\sum_{t=1}^{T}\eta_t .$$

Next, we use the definition $\eta_t = 1/(\lambda\, t)$ and note that the first sum on the right-hand side of the above equation collapses to $-\lambda\,(T+1)\,\|\mathbf{w}_{T+1} - \mathbf{u}\|^2 \le 0$. Thus,

$$\sum_{t=1}^{T} (\mathbb{E}[f(\mathbf{w}_t)] - f(\mathbf{u})) \le \frac{\rho^2}{2\,\lambda} \sum_{t=1}^{T} \frac{1}{t} \le \frac{\rho^2}{2\,\lambda}(1 + \ln(T)) \,.$$

The theorem follows from the above using Jensen's inequality. ■

(67577) Introduction to Machine Learning November 22, 2010

Lecture 7(a) – Stability and Regularization

Lecturer: Shai Shalev-Shwartz *Scribe: Shai Shalev-Shwartz*

Based on a book by Shai Ben-David and Shai Shalev-Shwartz (in preparation)

In this lecture we show that stable learning rules do not overfit. It follows that any hypothesis class for which we can find an algorithm which is both approximately ERM and stable is learnable. Finally, we show how regularization leads to stability.

14 Stable rules do not overfit

Let A be a learning algorithm and denote by $A(S)$ its output hypothesis on a training sequence $S = (\mathbf{z}_1, \ldots, \mathbf{z}_m)$. Let $\mathbf{z}'$ be another example. For each i, let $S^{(i)}$ be the training sequence in which we *replace* example $\mathbf{z}_i$ with $\mathbf{z}'$, namely, $S^{(i)} = (\mathbf{z}_1, \ldots, \mathbf{z}_{i-1}, \mathbf{z}', \mathbf{z}_{i+1}, \ldots, \mathbf{z}_m)$.

Definition 1 *We say that A is ϵ-replace-one-stable if for any S, i, and z' we have*

$$\ell(h_S; \mathbf{z}') - \ell(h_{S^{(i)}}; \mathbf{z}') \;\le\; \epsilon \,.$$

In words, A is replace-one-stable if a replacement of $\mathbf{z}_i$ by $\mathbf{z}'$ does not change the prediction of the output of A on $\mathbf{z}'$ by more than ϵ.

The following theorem shows that stable algorithms do not overfit.

Theorem 6 *Let A be an ϵ-replace-one-stable algorithm. Then,*

$$\mathbb{E}_{S\sim\mathcal{D}^m}[L_{\mathcal{D}}(A(S)) - L_S(A(S))] \;\le\; \epsilon \,.$$

Proof To simplify notation, denote $h_S = A(S)$. To prove the theorem, we will show below that for two i.i.d. sequences of examples $S = (\mathbf{z}_1, \ldots, \mathbf{z}_m)$ and $S' = (\mathbf{z}'_1, \ldots, \mathbf{z}'_m)$ we have

$$\mathbb{E}_{S\sim\mathcal{D}}[L_{\mathcal{D}}(h_S) - L_S(h_S)] = \frac{1}{m}\sum_{i=1}^{m} \mathbb{E}_{S,S'\sim\mathcal{D}}[\ell(h_S, z'_i) - \ell(h_{S^{(i)}}, z'_i)] \,. \tag{5}$$

Indeed, for any i, since $\mathbf{z}_i$ and $\mathbf{z}'_i$ are both drawn i.i.d. from $\mathcal{D}$, we have that

$$\mathbb{E}_{S\sim\mathcal{D}^m}\left[\ell(h_S; \mathbf{z}_i)\right] = \mathbb{E}_{S\sim\mathcal{D}^m, \mathbf{z}'_i\sim\mathcal{D}}\left[\ell(h_{S^{(i)}}; \mathbf{z}'_i)\right].$$

Hence,

$$\begin{aligned}\mathop{\mathbb{E}}_{S\sim\mathcal{D}^m}[L_S(h_S)] &= \mathop{\mathbb{E}}_{S\sim\mathcal{D}^m}\left[\frac{1}{m}\sum_{i=1}^m \ell(h_S;\mathbf{z}_i)\right] \\ &= \frac{1}{m}\sum_{i=1}^m \mathop{\mathbb{E}}_{S\sim\mathcal{D}^m}[\ell(h_S;\mathbf{z}_i)] \\ &= \frac{1}{m}\sum_{i=1}^m \mathop{\mathbb{E}}_{S\sim\mathcal{D}^m,\mathbf{z}'_i\sim\mathcal{D}}[\ell(h_{S^{(i)}};\mathbf{z}'_i)] \ .\end{aligned}$$

Also note that $L_{\mathcal{D}}(h_S) = \mathbb{E}_{\mathbf{z}'_i\sim\mathcal{D}}[\ell(h_S;\mathbf{z}'_i)] = \frac{1}{m}\sum_{i=1}^m \mathbb{E}_{\mathbf{z}'_i\sim\mathcal{D}}[\ell(h_S;\mathbf{z}'_i)]$. This yields Eq. (5). Finally, the assumption that the ERM rule is stable implies that the right-hand side of Eq. (5) is at most ϵ which concludes our proof. ■

The next theorem shows that if an algorithm is both stable and is an approximately ERM algorithm, then it approximately learns $\mathcal{H}$ (in expectation over the training set).

Theorem 7 *Let $\mathcal{H}$ be a hypothesis class, let $\mathcal{D}$ be a distribution over a domain Z, and let $h^\star \in \mathcal{H}$ be an arbitrary hypothesis. Let A be an ϵ_1-replace-one-stable learning rule that returns a hypothesis h_S which is ϵ_2-approximately ERM, in particular,*

$$L_S(h_S) \le L_S(h^\star) + \epsilon_2 \ .$$

Then,

$$\mathop{\mathbb{E}}_{S\sim\mathcal{D}^m}[L_{\mathcal{D}}(h_S)] - L_{\mathcal{D}}(h^\star) \ \le\ \epsilon_1 + \epsilon_2 \ .$$

Proof Since $\mathbb{E}[L_S(h_S) - L_S(h^\star)] \le \epsilon_2$ and $\mathbb{E}[L_S(h^\star) - L_{\mathcal{D}}(h^\star)] = 0$ we have

$$\begin{aligned}\mathbb{E}[L_{\mathcal{D}}(h_S) - L_{\mathcal{D}}(h^\star)] &= \mathbb{E}[L_{\mathcal{D}}(h_S) - L_S(h_S) + L_S(h_S) - L_S(h^\star) + L_S(h^\star) - L_{\mathcal{D}}(h^\star)] \\ &\le \mathbb{E}[L_{\mathcal{D}}(h_S) - L_S(h_S)] + \epsilon_2 + 0 \ .\end{aligned}$$

Combining the above with Theorem 6 concludes our proof. ■

15 Obtaining stability by Regularization

In the previous lecture we discussed convex-Lipschitz learning problems in which the loss function $\ell(\mathbf{w},\mathbf{z})$ is convex and Lipschitz w.r.t. $\mathbf{w}$. We saw that such problems are learnable using Stochastic Gradient Descent. We now show another route for learning such problems which is based on regularization and stability. The idea is to use a *regularized loss minimization* approach, in which we minimize the empirical loss plus a penalty term which aims at keeping the norm of the vector $\mathbf{w}$ to be small. The following theorem guarantees that this scheme gives a stable algorithm.

Theorem 8 *Let $\ell(\mathbf{w},\mathbf{z})$ be a loss function which is convex and ρ-Lipschitz w.r.t. its first argument. Given $\lambda > 0$, consider the learning rule:*

$$A(S) = \operatorname*{argmin}_{\mathbf{w}} L_S(\mathbf{w}) + \lambda\|\mathbf{w}\|^2 \ .$$

Then, A is ϵ-replace-one-stable with $\epsilon = \frac{2\rho^2}{\lambda\, m}$.

Proof To simplify our notation, denote $\mathbf{w} = A(S)$ and $\mathbf{w}' = A(S^{(i)})$, where $S^{(i)}$ is obtained by replacing example $\mathbf{z}_i$ of S with example $\mathbf{z}'$. We have,

$$\begin{aligned}
&L_S(\mathbf{w}') + \lambda\|\mathbf{w}'\|^2 - (L_S(\mathbf{w}) + \lambda\|\mathbf{w}\|^2) \\
&\qquad = L_S(\mathbf{w}') - L_S(\mathbf{w}) + \lambda(\|\mathbf{w}'\|^2 - \|\mathbf{w}\|^2) \\
&\qquad = L_{S'}(\mathbf{w}') - L_{S'}(\mathbf{w}) + \frac{\ell(\mathbf{w}',\mathbf{z}_i) - \ell(\mathbf{w},\mathbf{z}_i)}{m} - \frac{\ell(\mathbf{w}',\mathbf{z}') - \ell(\mathbf{w},\mathbf{z}')}{m} + \lambda(\|\mathbf{w}'\|^2 - \|\mathbf{w}\|^2) \\
&\qquad \le \frac{|\ell(\mathbf{w}',\mathbf{z}_i) - \ell(\mathbf{w},\mathbf{z}_i)|}{m} - \frac{|\ell(\mathbf{w}',\mathbf{z}') - \ell(\mathbf{w},\mathbf{z}')|}{m} \\
&\qquad \le \frac{2\,\rho}{m}\|\mathbf{w}' - \mathbf{w}\|\ ,
\end{aligned}$$

where the first inequality follows from the definition of $\mathbf{w}'$ as minimizer of $L_{S'}(\mathbf{v}) + \lambda\|\mathbf{v}\|^2$ and the second inequality follows from the assumption that ℓ is ρ-Lipschitz. Next, we use the fact that the function $f(\mathbf{v}) = L_S(\mathbf{v}) + \lambda\|\mathbf{v}\|^2$ is 2λ-strongly-convex and the minimality of $\mathbf{w}$ to get that

$$L_S(\mathbf{w}') + \lambda\|\mathbf{w}'\|^2 - (L_S(\mathbf{w}) + \lambda\|\mathbf{w}\|^2) \ge \lambda\|\mathbf{w}' - \mathbf{w}\|^2\ .$$

Combining the two inequalities we get

$$\|\mathbf{w}' - \mathbf{w}\| \le \frac{2\rho}{\lambda\, m}\ .$$

Combining this with the Lipschitz assumption on ℓ we conclude that

$$|\ell(\mathbf{w},\mathbf{z}') - \ell(\mathbf{w}',\mathbf{z}')| \le \rho\|\mathbf{w}' - \mathbf{w}\| \le \frac{2\rho^2}{\lambda\, m}\ ,$$

which concludes our proof. ■

As a corollary, we obtain the following.

Theorem 9 *Let $\ell(\mathbf{w},\mathbf{z})$ be a loss function which is convex and ρ-Lipschitz w.r.t. its first argument. Given $\lambda > 0$, consider the learning rule:*

$$A(S) = \operatorname*{argmin}_{\mathbf{w}} L_S(\mathbf{w}) + \lambda\|\mathbf{w}\|^2\ .$$

Then, for any $\mathbf{w}^\star$,

$$\mathop{\mathbb{E}}_{S}[L_{\mathcal{D}}(A(S)] - L_{\mathcal{D}}(\mathbf{w}^\star)\ \le\ \frac{2\rho^2}{\lambda\, m} + \lambda\|\mathbf{w}^\star\|^2\ .$$

Proof Denote $\mathbf{w} = A(S)$. For any $\mathbf{w}$ we have

$$L_S(\mathbf{w}) \le L_S(\mathbf{w}) + \lambda\|\mathbf{w}\|^2 \le L_S(\mathbf{w}^\star) + \lambda\|\mathbf{w}^\star\|^2$$

Combining the above, Theorem 8, and Theorem 7, we conclude our proof. ■

(67577) Introduction to Machine Learning November 22, 2010

Lecture 7(b) – SVM

Lecturer: Shai Shalev-Shwartz *Scribe: Shai Shalev-Shwartz*

Based on a book by Shai Ben-David and Shai Shalev-Shwartz (in preparation)

In this lecture we discuss the Support Vector Machine (SVM) method. SVM is one of the most popular machine learning algorithms. It combines many of the ideas we have already encountered: separation with margin, the hinge-loss, regularization, and kernels.

16 Hard-SVM

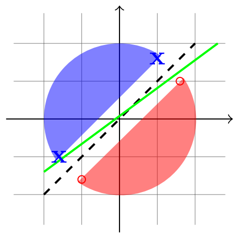


Figure 4: Both the black and green hyperplanes separate the four examples. The black hyperplane separates with the largest margin.

Let S be a linearly separable training set. There are many hyperplanes that can separate the data (see Figure 4). Which one of them should we choose? One option we have discussed previously is choosing the output of the Perceptron algorithm, since it depends on at most $1/\gamma(S)^2$ examples, and therefore we could apply the compression bound. In Hard-SVM we choose the hyperplane which separates the training set with the largest margin (see again Figure 4). Formally, this is the vector which achieves the maximum in the definition of $\gamma(S)$, that is,

$$\mathbf{w}^\star = \operatorname*{argmax}_{\mathbf{w}:\|\mathbf{w}\|_2=1} \min_{(\mathbf{x},y)\in S} y\langle \mathbf{w},\mathbf{x}\rangle \ .$$

The lemma below shows that $\mathbf{w}^\star$ can be obtained by solving a quadratic optimization problem (hence, can be found efficiently).

Lemma 8 *Define*

$$\mathbf{w}_0 = \arg\min\{\|\mathbf{w}\|^2 \ : \ \forall(\mathbf{x},y)\in S,\ y\langle \mathbf{w},\mathbf{x}\rangle \ge 1\} \ . \tag{6}$$

Then,

$$\frac{\mathbf{w}_0}{\|\mathbf{w}_0\|} = \operatorname*{argmax}_{\mathbf{w}:\|\mathbf{w}\|_2=1} \min_{(\mathbf{x},y)\in S} y\langle \mathbf{w},\mathbf{x}\rangle \ ,$$

that is, the normalization of $\mathbf{w}_0$ corresponds to the largest margin separating hyperplane.

Proof Define $\hat{\mathbf{w}} = \frac{\mathbf{w}_0}{\|\mathbf{w}_0\|}$ and let $\mathbf{w}^\star$ be the largest-margin separating hyperplane. We need to show that $\hat{\mathbf{w}} = \mathbf{w}^\star$. To do so, first note that the vector $\frac{\mathbf{w}^\star}{\gamma(S)}$ satisfies the constraints of the optimization problem given in Eq. (6) and therefore $\|\mathbf{w}_0\| \le \|\mathbf{w}^\star/\gamma(S)\| = 1/\gamma(S)$. Therefore, for all $(\mathbf{x},y)\in S$

$$y_i\langle \hat{\mathbf{w}},\mathbf{x}_i\rangle = \frac{1}{\|\mathbf{w}_0\|} y_i\langle \mathbf{w}_0,\mathbf{x}_i\rangle \ge \frac{1}{\|\mathbf{w}_0\|} \ge \gamma(S) \ .$$

But since $\|\hat{\mathbf{w}}\| = 1$ the above implies that $\hat{\mathbf{w}} = \mathbf{w}^\star$. ■

16.1 Optimality conditions and ”Support Vectors”

The name ”Support Vector Machine” stems from the fact that $\mathbf{w}^\star$ is supported by (i.e. is in the linear span of) the examples that are exactly at distance $1/\|\mathbf{w}^\star\|$ from the separating hyperplane. These vectors are therefore called *support vectors*. To see this, we rely on **Fritz John optimality conditions**.

Definition 2 (Fritz John) *Consider the problem:*

$$\min_{\mathbf{w}} f(\mathbf{w}) \quad \textit{s.t.} \ \ \forall i \in [m], \ g_i(\mathbf{w}) \le 0 \,,$$

where $f, g_1, \ldots, g_m$ are differentiable. Then, if $\mathbf{w}^\star$ is an optimal solution then there exists $\boldsymbol{\alpha} \in \mathbb{R}^m$ such that $\nabla f(\mathbf{w}^\star) + \sum_{i \in I} \alpha_i \nabla g_i(\mathbf{w}^\star) = \mathbf{0}$, where $I = \{i : g_i(\mathbf{w}^\star) = 0\}$.

Applying Fritz John condition on Eq. (6) we obtain:

Corollary 5 *Let $\mathbf{w}^\star$ be a minimizer of Eq. (6) and let $I = \{i : \langle \mathbf{w}^\star, \mathbf{x}_i \rangle = 1\}$. Then, there exist coefficients α_i such that*

$$\mathbf{w}^\star = \sum_{i \in I} \alpha_i \mathbf{x}_i \,.$$

16.2 Duality

Traditionally, many of the properties we will derive for SVM have been obtained by switching to the dual problem. For completeness, we present below how to derive the dual of Eq. (6). We start by rewriting the problem in an equivalent form as follows. Consider the function

$$g(\mathbf{w}) \;=\; \max_{\boldsymbol{\alpha} \in \mathbb{R}^m : \boldsymbol{\alpha} \ge \mathbf{0}} \sum_{i=1}^m \alpha_i (1 - y_i \langle \mathbf{w}, \mathbf{x}_i \rangle) \;=\; \begin{cases} 0 & \text{if } \forall i, y_i \langle \mathbf{w}, \mathbf{x}_i \rangle \ge 1 \\ \infty & \text{otherwise} \end{cases} \,.$$

We can therefore rewrite Eq. (6) as

$$\min_{\mathbf{w}} \|\mathbf{w}\|_2^2 + g(\mathbf{w}) \,. \tag{7}$$

Rearranging the above we obtain

$$\min_{\mathbf{w}} \max_{\boldsymbol{\alpha} \in \mathbb{R}^m : \boldsymbol{\alpha} \ge \mathbf{0}} \frac{1}{2} \|\mathbf{w}\|_2^2 + \sum_{i=1}^m \alpha_i (1 - y_i \langle \mathbf{w}, \mathbf{x}_i \rangle) \,. \tag{8}$$

We have therefore shown that the above problem is equivalent to the hard SVM problem given in Eq. (6). Now suppose that we flip the order of $\min$ and $\max$ in the above equation. It is easy to verify that this can only decrease the objective value, namely,

$$\min_{\mathbf{w}} \max_{\boldsymbol{\alpha} \in \mathbb{R}^m : \boldsymbol{\alpha} \ge \mathbf{0}} \frac{1}{2} \|\mathbf{w}\|_2^2 + \sum_{i=1}^m \alpha_i (1 - y_i \langle \mathbf{w}, \mathbf{x}_i \rangle) \;\ge\; \max_{\boldsymbol{\alpha} \in \mathbb{R}^m : \boldsymbol{\alpha} \ge \mathbf{0}} \min_{\mathbf{w}} \frac{1}{2} \|\mathbf{w}\|_2^2 + \sum_{i=1}^m \alpha_i (1 - y_i \langle \mathbf{w}, \mathbf{x}_i \rangle) \,.$$

The above inequality is called *weak duality*. It turns out that in our case, *strong duality* also holds, namely, the above inequality holds with equality. The right-hand side is called the *dual* problem, namely,

$$\max_{\boldsymbol{\alpha} \in \mathbb{R}^m : \boldsymbol{\alpha} \ge \mathbf{0}} \min_{\mathbf{w}} \frac{1}{2} \|\mathbf{w}\|_2^2 + \sum_{i=1}^m \alpha_i (1 - y_i \langle \mathbf{w}, \mathbf{x}_i \rangle) \,. \tag{9}$$

We can rewrite the dual problem by noting that once $\boldsymbol{\alpha}$ is fixed, the optimization problem w.r.t. $\mathbf{w}$ is unconstrained and the objective is differentiable, thus, at the optimum, the gradient equals to zero:

$$\mathbf{w} - \sum_{i=1}^m \alpha_i y_i \mathbf{x}_i \;=\; 0 \quad \Rightarrow \quad \mathbf{w} = \sum_{i=1}^m \alpha_i y_i \mathbf{x}_i \,.$$

This shows us that the solution must be in the linear span of the examples, a fact we will use later to derive SVM with kernels. Plugging the above into Eq. (9) we obtain that the dual problem can be rewritten as

$$\max_{\boldsymbol{\alpha}\in\mathbb{R}^m:\boldsymbol{\alpha}\geq\mathbf{0}} \frac{1}{2}\left\|\sum_{i=1}^m \alpha_i y_i \mathbf{x}_i\right\|_2^2 + \sum_{i=1}^m \alpha_i(1 - y_i\langle\sum_j \alpha_j y_j \mathbf{x}_j, \mathbf{x}_i\rangle) \ . \tag{10}$$

Rearranging the above gives the problem

$$\max_{\boldsymbol{\alpha}\in\mathbb{R}^m:\boldsymbol{\alpha}\geq\mathbf{0}} \sum_{i=1}^m \alpha_i - \frac{1}{2}\sum_{i=1}^m\sum_{j=1}^m \alpha_i\alpha_j y_i y_j \langle \mathbf{x}_j, \mathbf{x}_i\rangle \ . \tag{11}$$

Note that the dual problem only involves inner products between vectors in the feature space and does not require direct access to specific elements of the feature space. This enables to use SVM with kernels. We will elaborate on this in later sections.

16.3 Bias term

SVM can be defined with non-homogenous hyperplane. That is, the classification rule is based on $\text{sign}(\langle\mathbf{w},\mathbf{x}\rangle + b)$, where b is a scalar. For simplicity, we sticked to the homogenous hyperplane formulation.

17 Soft-SVM

The Hard-SVM formulation assumes that the training set is linearly separable. This is a rather strong assumption. A natural relaxation is to maximize the margin while minimizing the number of examples that violate the margin constraint. However, this leads to a non-convex optimization problem which is hard to solve. As we have shown in previous lectures, one way to circumvent the hardness result is by relying on surrogate convex loss function. Indeed, SVM relies on the hinge-loss function. The Sotf-SVM optimization problem is therefore

$$\min_{\mathbf{w}} \lambda\|\mathbf{w}\|^2 + \frac{1}{|S|}\sum_{(\mathbf{x},y)\in S} \max\{0, 1 - y\langle\mathbf{w},\mathbf{x}\rangle\} \ , \tag{12}$$

where λ is a parameter that controls the tradeoff between large margin (i.e. small norm of $\mathbf{w}$) and fit to the data (i.e. small hinge-loss).

17.1 Analyzing Sotf-SVM using stability

The loss function $\ell(\mathbf{w},(\mathbf{x},y))$ is convex and 1-Lipschitz w.r.t. its first argument. Therefore, using Theorem 9 we obtain that for any $\mathbf{u}$, the solution of SVM, denoted $A(S)$ satisfies

$$\mathop{\mathbb{E}}_S[L_{\mathcal{D}}(A(S)] \ \leq \ L_{\mathcal{D}}(\mathbf{u}) + \lambda\|\mathbf{u}\|^2 + \frac{2\rho^2}{\lambda\, m} \ ,$$

where $L_{\mathcal{D}}$ is the expected hinge-loss. Note that this bound is similar to the oracle inequalities we derived using the MDL bound, with $\|u\|$ taking the role of the description length of $\mathbf{u}$.

18 SVM with kernels

Recall that in previous lectures we showed how to run the Perceptron algorithm with kernels. The idea was to run the Perceptorn with feature vector $\psi(\mathbf{x}_i)$, where the range of ψ can be very large, while only accessing a kernel function that implements inner products in the feature space. We did it by showing that the Perceptron algorithm can be run while only calculating inner products, $\langle \psi(\mathbf{x}), \psi(\mathbf{x}') \rangle$, without ever explicitly accessing $\psi(\mathbf{x})$. We now show that the kernel trick also works for SVM. To do so, first note that both soft and hard SVM optimization problems are special cases of a general optimization problem of the form:

$$\min_{\mathbf{w}} f(y_1 \langle \mathbf{w}, \psi(\mathbf{x}_1) \rangle, \ldots, y_m \langle \mathbf{w}, \psi(\mathbf{x}_m) \rangle) + R(\|\mathbf{w}\|_2) \ , \tag{13}$$

where $f : \mathbb{R}^m \to \mathbb{R}$ is an arbitrary function and $R : \mathbb{R}_+ \to \mathbb{R}$ is a monotonically non-decreasing function. The following theorem tells us that there exists an optimal solution of Eq. (13) that lies in the span of the examples.

Theorem 10 (Wahba's Representer Theorem) *Assume that ψ is a mapping from $\mathcal{X}$ to a Hilbert space. Then, there exists a vector $\boldsymbol{\alpha} \in \mathbb{R}^m$ such that $\mathbf{w} = \sum_{i=1}^m \alpha_i \psi(\mathbf{x}_i)$ is an optimal solution of Eq. (13).*

Proof Let $\mathbf{w}^\star$ be an optimal solution of Eq. (13). Because $\mathbf{w}^\star$ is an element of a Hilbert space, we can rewrite $\mathbf{w}^\star$ as

$$\mathbf{w}^\star = \sum_{i=1}^m \alpha_i \psi(\mathbf{x}_i) + \mathbf{u} \ ,$$

where $\langle \mathbf{u}, \psi(\mathbf{x}_i) \rangle = 0$ for all i. Set $\mathbf{w} = \mathbf{w}^\star - \mathbf{u}$. Clearly, $\|\mathbf{w}^\star\|_2^2 = \|\mathbf{w}\|_2^2 + \|\mathbf{u}\|_2^2$, thus $\|\mathbf{w}\|_2 \leq \|\mathbf{w}^\star\|_2$. Since R is non-decreasing we obtain that $R(\|\mathbf{w}\|) \leq R(\|\mathbf{w}^\star\|)$. Additionally, for all i we have that

$$y_i \langle \mathbf{w}, \psi(\mathbf{x}_i) \rangle = y_i \langle \mathbf{w} + \mathbf{u}, \psi(\mathbf{x}_i) \rangle = y_i \langle \mathbf{w}^\star, \psi(\mathbf{x}_i) \rangle \ ,$$

hence the value of f for $\mathbf{w}^\star$ and $\mathbf{w}$ will be the same. We have shown that the objective of Eq. (13) at $\mathbf{w}$ cannot be larger than the objective at $\mathbf{w}^\star$ and therefore $\mathbf{w}$ is also an optimal solution, which concludes our proof. ■

Based on the representer theorem we can optimize Eq. (13) w.r.t. the coefficients $\boldsymbol{\alpha}$ instead of the coefficients $\mathbf{w}$ as follows. Writing $\mathbf{w} = \sum_{j=1}^m \alpha_j \psi(\mathbf{x}_j)$ we have that for all i

$$\langle \mathbf{w}, \psi(\mathbf{x}_i) \rangle = \langle \sum_j \alpha_j \psi(\mathbf{x}_j), \psi(\mathbf{x}_i) \rangle = \sum_{j=1}^m \alpha_j \langle \psi(\mathbf{x}_j), \psi(\mathbf{x}_i) \rangle \ .$$

Similarly,

$$\|\mathbf{w}\|_2^2 = \langle \sum_j \alpha_j \psi(\mathbf{x}_j), \sum_j \alpha_j \psi(\mathbf{x}_j) \rangle = \sum_{i,j=1}^m \alpha_i \alpha_j \langle \psi(\mathbf{x}_i), \psi(\mathbf{x}_j) \rangle \ .$$

Let $K(\mathbf{x}, \mathbf{x}') = \langle \psi(\mathbf{x}), \psi(\mathbf{x}') \rangle$ be a kernel function that implements inner products in the feature space. Instead of solving Eq. (13) we can solve the equivalent problem

$$\min_{\boldsymbol{\alpha} \in \mathbb{R}^m} f\left(y_1 \sum_{j=1}^m \alpha_j K(\mathbf{x}_j, \mathbf{x}_1), \ldots, y_m \sum_{j=1}^m \alpha_j K(\mathbf{x}_j, \mathbf{x}_m) \right) + R\left(\sqrt{\sum_{i,j=1}^m \alpha_i \alpha_j K(\mathbf{x}_j, \mathbf{x}_i)} \right) \ . \tag{14}$$

To solve the optimization problem given in Eq. (14), we do not need any direct access to elements in the features space. The only thing we should know is how to calculate inner-products in the feature space, or equivalently, to calculate the kernel function. In fact, to solve Eq. (14) we solely need to know the value of the $m \times m$ matrix G s.t. $G_{i,j} = K(\mathbf{x}_i, \mathbf{x}_j)$, which is often called the *Gram* matrix.

Once we learned the coefficients $\boldsymbol{\alpha}$ we can calculate the prediction on a new instance by:

$$\langle \mathbf{w}, \psi(\mathbf{x}) \rangle = \sum_{j=1}^m \alpha_j \langle \psi(\mathbf{x}_j), \psi(\mathbf{x}) \rangle = \sum_{j=1}^m \alpha_j K(\mathbf{x}_j, \mathbf{x}) \ .$$

19 Solving Soft-SVM with Stochastic Gradient Descent

In Algorithm 6 we showed a general scheme for minimizing strongly convex functions using stochastic gradient descent. We can rewrite the Soft-SVM objective function as

$$f(\mathbf{w}) = \mathop{\mathbb{E}}_{(\mathbf{x},y)\sim S}\left[\lambda\|\mathbf{w}\|^2 + \max\{0, 1 - y\langle \mathbf{w}, \mathbf{x}\rangle\}\right] . \tag{15}$$

From this fact, we obtain the following algorithm for solving Soft-SVM.

Algorithm 7 Stochastic sub-gradient descent for solving Soft-SVM

```
**Goal:** Solve $\min_{\mathbf{w}} f(\mathbf{w})$ where $f(\mathbf{w})$ is as in Eq. (15)
**Initialize:** $\mathbf{w}_1 = \mathbf{0}$
**for** $t = 1, \ldots, T$
  Choose example $(\mathbf{x}, y)$ uniformly at random from $S$
  If $y(\langle \mathbf{w}_t, \mathbf{x}\rangle < 1)$
    Set $\mathbf{w}_{t+1} = (1 - \frac{1}{t})\mathbf{w}_t + \frac{1}{\lambda\, t} y\mathbf{x}$
  Else
    Set $\mathbf{w}_{t+1} = (1 - \frac{1}{t})\mathbf{w}_t$
**end for**
**Output:** $\bar{\mathbf{w}} = \frac{1}{T}\sum_{t=1}^{T} \mathbf{w}_t$
```

(67577) Introduction to Machine Learning November 22, 2010

Lecture 7(c) – Ridge Regression

Lecturer: Shai Shalev-Shwartz *Scribe: Shai Shalev-Shwartz*

Based on a book by Shai Ben-David and Shai Shalev-Shwartz (in preparation)

Lecture 11 from 2009.

(67577) Introduction to Machine Learning November 29, 2010

Lecture 8 – VC dimension and No Free Lunch

Lecturer: Shai Ben-David *Scribe: Shai Shalev-Shwartz*

Based on a book by Shai Ben-David and Shai Shalev-Shwartz (in preparation)

Lecture 6 from 2009.

20 The main theorem of statistical learning

Theorem 11 (Vapnik-Chervonenkis 1971, Blumer-Ehrenfeucht-Hausler-Warmuth 1989) *Let $\mathcal{H}$ be a class of binary classifiers. The following are equivalent:*

1. *$\mathcal{H}$ has the uniform convergence property*

2. *The ERM generalizes*
3. *$\mathcal{H}$ is agnostic-PAC learnable*
4. *$\mathcal{H}$ is PAC learnable*
5. $\mathrm{VCdim}(\mathcal{H}) < \infty$

Proof $1 \to 2 \to 3 \to 4$ is trivial. $4 \to 5$ follows from the No-Free-Lunch theorem. The difficult part is to show that $5 \to 1$. In the next section we show that $5 \to 4$ using ϵ-nets. A similar technique can show that $5 \to 1$. ■

21 ϵ-nets

Definition 3 (ϵ-net) *Let $\mathcal{X}$ be a domain. $S \subset \mathcal{X}$ is ϵ-net for $H \subset 2^{\mathcal{X}}$ with respect to a distribution $\mathcal{D}$ over $\mathcal{X}$ if*

$$\forall h \in \mathcal{H}: \ \mathcal{D}(h) \geq \epsilon \ \Rightarrow \ h \cap S \neq \emptyset .$$

Theorem 12 *Let $\mathcal{H} \subset 2^{\mathcal{X}}$ with $\mathrm{VCdim}(\mathcal{H}) = d$. Fix $\epsilon \in (0,1)$, $\delta \in (0, 1/4)$ and let*

$$m \geq \frac{8}{\epsilon}\left(2d \log\left(\frac{16e}{\epsilon}\right) + \log\left(\frac{2}{\delta}\right)\right) .$$

Then, with probability of at least $1 - \delta$ over a choice of $S \sim \mathcal{D}^m$ we have that S is an ϵ-net for $\mathcal{H}$.

Proof Let

$$B = \{S \subset \mathcal{X} \ : \ |S| = m, \ \exists h \in \mathcal{H}, \mathcal{D}(h) \geq \epsilon, h \cap S = \emptyset\}$$

be the set of sets which are not ϵ-nets. We need to bound $\mathbb{P}[S \in B]$. Define

$$B' = \{(S,T) \subset \mathcal{X} \ : \ |S| = |T| = m, \ \exists h \in \mathcal{H}, \mathcal{D}(h) \geq \epsilon, h \cap S = \emptyset, \ |T \cap h| > \tfrac{\epsilon m}{2}\} .$$

Claim 1 $\mathbb{P}[S \in B] \leq 2\,\mathbb{P}[(S,T) \in B']$.

Proof of Claim 1: Since S and T are chosen independently we can write

$$\mathbb{P}[(S,T) \in B'] = \mathop{\mathbb{E}}_{(S,T)\sim\mathcal{D}^{2m}}\left[\mathbb{1}_{[(S,T)\in B']}\right] = \mathop{\mathbb{E}}_{S\sim\mathcal{D}^m}\left[\mathop{\mathbb{E}}_{T\sim\mathcal{D}^m}\left[\mathbb{1}_{[(S,T)\in B']}\right]\right] .$$

Note that $(S,T) \in B'$ implies $S \in B$ and therefore $\mathbb{1}_{[(S,T)\in B']} = \mathbb{1}_{[(S,T)\in B']}\,\mathbb{1}_{[S\in B]}$ which gives

$$\begin{aligned}\mathbb{P}[(S,T) \in B'] &= E_{S\sim\mathcal{D}^m} \mathop{\mathbb{E}}_{T\sim\mathcal{D}^m} \mathbb{1}_{[(S,T)\in B']}\,\mathbb{1}_{[S\in B]} \\ &= \mathop{\mathbb{E}}_{S\sim\mathcal{D}^m} \mathbb{1}_{[S\in B]} \mathop{\mathbb{E}}_{T\sim\mathcal{D}^m} \mathbb{1}_{[(S,T)\in B']} .\end{aligned}$$

Fix some S. Then, either $\mathbb{1}_{[S\in B]} = 0$ or $S \in B$ and then $\exists h_S$ such that $\mathcal{D}(h_S) \geq \epsilon$ and $|h_S \cap S| = 0$. It follows that a sufficient condition for $(S,T) \in B'$ is that $|T \cap h_S| > \frac{\epsilon m}{2}$. Therefore, whenever $S \in B$ we have

$$\mathop{\mathbb{E}}_{T\sim\mathcal{D}^m} \mathbb{1}_{[(S,T)\in B']} \ \geq \ \mathop{\mathbb{P}}_{T\sim\mathcal{D}^m}[|T \cap h_S| > \tfrac{\epsilon m}{2}] .$$

But, since we now assume $S \in B$ we know that $\mathcal{D}(h_S) = \rho \geq \epsilon$. Therefore, $|T \cap h_S|$ is a binomial random variable with parameters ρ (probability of success for a single try) and m (number of tries). Chernoff's inequality implies

$$\mathbb{P}[|T \cap h_S| \leq \tfrac{\rho m}{2}] \ \leq \ e^{-\frac{2}{m\rho}(m\rho - m\rho/2)^2} = e^{-m\rho/2} \leq e^{-m\epsilon/2} \leq e^{-d\log(1/\delta)/2} = \delta^{d/2} \leq 1/2 .$$

Thus,

$$\mathbb{P}[|T \cap h_S| > \tfrac{\epsilon m}{2}] \ = \ 1 - \mathbb{P}[|T \cap h_S| \leq \tfrac{\epsilon m}{2}] \ \geq \ 1 - \mathbb{P}[|T \cap h_S| \leq \tfrac{\rho m}{2}] \ \geq \ 1/2 .$$

Combining all the above we *conclude the proof of Claim 1.*

Claim 2 (Symmetrization): $\mathbb{P}[(S,T) \in B'] \le e^{-\epsilon m/4}\, \tau_{\mathcal{H}}(2m)$.

Proof of Claim 2: To simplify notation, let $\alpha = m\epsilon/2$ and for a sequence $A = (x_1, \ldots, x_{2m})$ let $A_0 = (x_1, \ldots, x_m)$. Using the definition of B' we get that

$$\begin{aligned}\mathbb{P}[A \in B'] &= \mathop{\mathbb{E}}_{A\sim\mathcal{D}^{2m}} \max_{h\in\mathcal{H}} \mathbb{1}_{[\mathcal{D}(h)\ge\epsilon]}\, \mathbb{1}_{[|h\cap A_0|=0]}\, \mathbb{1}_{[|h\cap A|\ge\alpha]} \\ &\le \mathop{\mathbb{E}}_{A\sim\mathcal{D}^{2m}} \max_{h\in\mathcal{H}} \mathbb{1}_{[|h\cap A_0|=0]}\, \mathbb{1}_{[|h\cap A|\ge\alpha]} \ .\end{aligned}$$

Now, let us define by $\mathcal{H}_A$ the effective number of different hypotheses on A, namely, $\mathcal{H}_A = \{h \cap A : h \in \mathcal{H}$. It follows that,

$$\begin{aligned}\mathbb{P}[A \in B'] &\le \mathop{\mathbb{E}}_{A\sim\mathcal{D}^{2m}} \max_{h\in\mathcal{H}_A} \mathbb{1}_{[|h\cap A_0|=0]}\, \mathbb{1}_{[|h\cap A|\ge\alpha]} \\ &\le \mathop{\mathbb{E}}_{A\sim\mathcal{D}^{2m}} \sum_{h\in\mathcal{H}_A} \mathbb{1}_{[|h\cap A_0|=0]}\, \mathbb{1}_{[|h\cap A|\ge\alpha]}\end{aligned}$$

Let $J = \{\mathbf{j} \subset [2m] : |\mathbf{j}| = m\}$. For any $\mathbf{j} \in J$ and $A = (x_1, \ldots, x_{2m})$ define $A_{\mathbf{j}} = (x_{j_1}, \ldots, x_{j_m})$. Since the elements of A are chosen i.i.d., we have that for any $\mathbf{j} \in J$ and any function $f(A, A_0)$ it holds that $\mathbb{E}_{A\sim\mathcal{D}^{2m}}[f(A,A_0)] = \mathbb{E}_{A\sim\mathcal{D}^{2m}}[f(A,A_{\mathbf{j}})]$. Since this holds for any $\mathbf{j}$ it also holds for the expectation of $\mathbf{j}$ chosen at random from J. In particular, it holds for the function $f(A,A_0) = \sum_{h\in\mathcal{H}_A} \mathbb{1}_{[|h\cap A_0|=0]}\, \mathbb{1}_{[|h\cap A|\ge\alpha]}$. We therefore obtain that,

$$\begin{aligned}\mathbb{P}[A \in B'] &\le \mathop{\mathbb{E}}_{A\sim\mathcal{D}^{2m}} \mathop{\mathbb{E}}_{j\sim J} \sum_{h\in\mathcal{H}_A} \mathbb{1}_{[|h\cap A_{\mathbf{j}}|=0]}\, \mathbb{1}_{[|h\cap A|\ge\alpha]} \\ &= \mathop{\mathbb{E}}_{A\sim\mathcal{D}^{2m}} \sum_{h\in\mathcal{H}_A} \mathbb{1}_{[|h\cap A|\ge\alpha]} \mathop{\mathbb{E}}_{j\sim J} \mathbb{1}_{[|h\cap A_{\mathbf{j}}|=0]} \ .\end{aligned}$$

Now, fix some A s.t. $|h \cap A| \ge \alpha$. Then, $\mathbb{E}_j\, \mathbb{1}_{[|h\cap A_{\mathbf{j}}|=0]}$ is the probability that when choosing m balls from a bag with at least α red balls, we will never choose a red ball. This probability is at most

$$(1 - \alpha/(2m))^m = (1 - \epsilon/4)^m \le e^{-\epsilon m/4} \ .$$

We therefore get that

$$\mathbb{P}[A \in B'] \le \mathop{\mathbb{E}}_{A\sim\mathcal{D}^{2m}} \sum_{h\in\mathcal{H}_A} e^{-\epsilon m/4} \le e^{-\epsilon m/4} \mathop{\mathbb{E}}_{A\sim\mathcal{D}^{2m}} |\mathcal{H}_A| \ .$$

Using the definition of the growth function we *conclude the proof of Claim 2.*

completing the proof: By Sauer's lemma we know that $\tau_{\mathcal{H}}(2m) \le (2em/d)^d$. Combining this with the two claims we obtain that

$$\mathbb{P}[S \in B] \le 2(2em/d)^d\, e^{-\epsilon m/4} \ .$$

We would like that the right-hand-side of the above will be at most δ, that is,

$$2(2em/d)^d\, e^{-\epsilon m/4} \le \delta \ .$$

Rearranging, we obtain the requirement

$$m \ge \frac{4}{\epsilon}\left(d\log(2em/d) + \log(2/\delta)\right) = \frac{4d}{\epsilon}\log(m) + \frac{4}{\epsilon}(d\log(2e/d) + \log(2/\delta) \ .$$

Using Lemma 10, a sufficient condition for the above to hold is that

$$m \ge \frac{16d}{\epsilon}\log\left(\frac{8d}{\epsilon}\right) + \frac{8}{\epsilon}(d\log(2e/d) + \log(2/\delta)$$

A sufficient condition for the above is that

$$\begin{aligned} m &\geq \frac{16d}{\epsilon}\log\left(\frac{8d}{\epsilon}\right)+\frac{16}{\epsilon}(d\log(2e/d)+\tfrac{1}{2}\log(2/\delta)) \\ &= \frac{16d}{\epsilon}\left(\log\left(\frac{8d\,2e}{d\epsilon}\right)\right)+\frac{8}{\epsilon}\log(2/\delta) \\ &= \frac{8}{\epsilon}\left(2d\log\left(\frac{16e}{\epsilon}\right)+\log\left(\frac{2}{\delta}\right)\right) . \end{aligned}$$

and this concludes our proof. ■

21.1 Two technical lemmas

Lemma 9 *Let $a > 0$. Then: $x \geq 2a\log(a) \ \Rightarrow \ x \geq a\log(x)$.*

Proof Consider the function $f(x) = x - a\log(x)$. The derivative is $f'(x) = 1 - a/x$. Thus, for $x > a$ the derivative is positive and the function in increases. In addition,

$$\begin{aligned} f(2a\log(a)) &= 2a\log(a) - a\log(2a\log(a)) \\ &= 2a\log(a) - a\log(a) - a\log(2\log(a)) \\ &= a\log(a) - a\log(2\log(a)) . \end{aligned}$$

Since $a - 2\log(a) > 0$ for all $a > 0$, the proof follows. ■

Lemma 10 *Let $a \geq 1$ and $b > 0$. Then: $x \geq 4a\log(2a) + 2b \ \Rightarrow \ x \geq a\log(x) + b$*

Proof It suffices to prove that $x \geq 4a\log(2a) + 2b$ implies that both $x \geq 2a\log(x)$ and $x \geq 2b$. Since we assume $a \geq 1$ we clearly have that $x \geq 2b$. In addition, since $b > 0$ we have that $x \geq 4a\log(2a)$ which using Lemma 9 implies that $x \geq 2a\log(x)$. This concludes our proof. ■

21.2 From ϵ-nets to PAC learnability

Theorem 13 *Let $\mathcal{H}$ be a hypothesis class over $\mathcal{X}$ with $\mathrm{VCdim}(\mathcal{H}) = d$. Let $\mathcal{D}$ be a distribution over $\mathcal{X}$ and let $c \in \mathcal{H}$ be a target hypothesis. Fix $\epsilon, \delta \in (0,1)$ and let m be as defined in Theorem 12. Then, with probability of at least $1-\delta$ over a choice of m i.i.d. instances from $\mathcal{X}$ with labels according to c we have that any ERM hypothesis has a generalization error of at most ϵ.*

Proof Define the class $\mathcal{H}^c = \{c \triangle h : h \in \mathcal{H}\}$, where $c \triangle h = (h \setminus c) \cup (c \setminus h)$. It is easy to verify that if some $A \subset \mathcal{X}$ is shattered by $\mathcal{H}$ then it is also shattered by $\mathcal{H}^c$ and vice versa. Hence, $\mathrm{VCdim}(\mathcal{H}) = \mathrm{VCdim}(\mathcal{H}^c)$. Therefore, using Theorem 12 we know that with probability of at least $1-\delta$, the sample S is an ϵ-net for $\mathcal{H}^c$. Note that $L_{\mathcal{D}}(h) = \mathcal{D}(h \triangle c)$. Therefore, for any $h \in \mathcal{H}$ with $L_{\mathcal{D}}(h) \geq \epsilon$ we have that $|(h \triangle c) \cap S| > 0$, which implies that h cannot be an ERM hypothesis, which concludes our proof. ■

(67577) Introduction to Machine Learning December 6, 2010

Lecture 9 – Boosting

Lecturer: Shai Shalev-Shwartz *Scribe: Shai Shalev-Shwartz*

Based on a book by Shai Ben-David and Shai Shalev-Shwartz (in preparation)

Lecture 18 from 2009.

(67577) Introduction to Machine Learning December 13, 2010

Lecture 10 – Nearest Neighbor

Lecturer: Shai Shalev-Shwartz *Scribe: Shai Shalev-Shwartz*

Based on a book by Shai Ben-David and Shai Shalev-Shwartz (in preparation)

Lecture 12 from 2009.

(67577) Introduction to Machine Learning December 20, 2010

Lecture 11 – Dimensionality Reduction

Lecturer: Shai Shalev-Shwartz *Scribe: Shai Shalev-Shwartz*

Based on a book by Shai Ben-David and Shai Shalev-Shwartz (in preparation)

Lecture 15 from 2009.

(67577) Introduction to Machine Learning December 27, 2010

Lecture 12 – Clustering

Lecturer: Shai Shalev-Shwartz *Scribe: Shai Shalev-Shwartz*

Based on a book by Shai Ben-David and Shai Shalev-Shwartz (in preparation)

Lecture 16 from 2009.

(67577) Introduction to Machine Learning January 3, 2010

Lecture 13 – Generative Models

Lecturer: Shai Shalev-Shwartz *Scribe: Shai Shalev-Shwartz*

Based on a book by Shai Ben-David and Shai Shalev-Shwartz (in preparation)

Lecture 17 from 2009.

(67577) Introduction to Machine Learning January 10, 2010

Lecture 14 – Online Learning

Lecturer: Shai Shalev-Shwartz *Scribe: Shai Shalev-Shwartz*

Based on a book by Shai Ben-David and Shai Shalev-Shwartz (in preparation)

Lecture 19 from 2009.